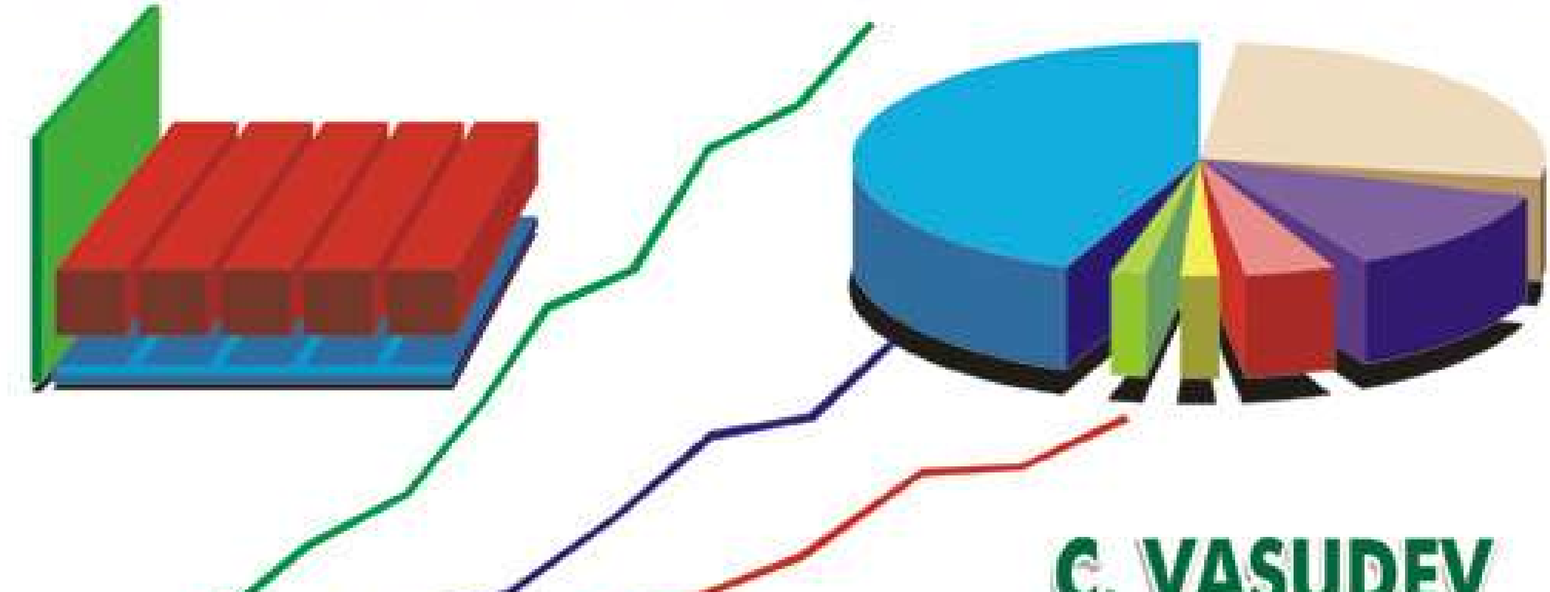
Graph Theory with Applications

(Common to all P.G. & U.G. Courses :
M.C.A.; M.C.S.; M.Tech.; M.I.T.; M.Sc.; B.C.A.; B.C.S.; B.I.T.; B.E.; B.Tech.; B.Sc.)

(COMMON TO ALL INDIAN UNIVERSITIES)



C. VASUDEV

NEW AGE INTERNATIONAL PUBLISHERS

Graph Theory with Application

This page intentionally left blank

Graph Theory with Application

(Common to all P.G. & U.G. Courses :
M.C.A.; M.C.S.; M.Tech.; M.I.T.; M.Sc.; B.C.A.; B.C.S.; B.I.T.; B.E.; B.Tech.; B.Sc.)

(COMMON TO ALL INDIAN UNIVERSITIES)

C. Vasudev
M.Sc., M.A.(Lit.), M.Phil., (M.Tech.) (IT)
Department of Mathematics
The Oxford Institutions, Bangalore, Karanataka

PUBLISHING FOR ONE WORLD

NEW AGE INTERNATIONAL (P) LIMITED, PUBLISHERS
New Delhi • Bangalore • Chennai • Cochin • Guwahati • Hyderabad
Jalandhar • Kolkata • Lucknow • Mumbai • Ranchi
Visit us at www.newagepublishers.com

Published by New Age International (P) Ltd., Publishers

ISBN : 978-81-224-2413-3

PUBLISHING FOR ONE WORLD

NEW AGE INTERNATIONAL (P) LIMITED, PUBLISHERS
4835/24, Ansari Road, Daryaganj, New Delhi - 110002
Visit us at **www.newagepublishers.com**

I
Dedicated this book
to
'Maha Kaali Maata'

This page intentionally left blank

PREFACE

This text has been carefully designed for flexible use. It is primarily designed to provide an introduction to some fundamental concepts in Graph Theory, for under-graduate and post-graduate students.

Each topic is divided into sections of approximately the same length, and each section is divided into subsections that form natural blocks of material for teaching. Instructors can easily pace their lectures using these blocks.

All definitions and theorems in this text are stated extremely carefully so that students will appreciate the precision of language and rigor needed in mathematical sciences. Proofs are motivated and developed slowly ; their steps are all carefully justified.

The writing style in this book is direct and pragmatic. Precise mathematical language is used without excessive formalism and abstraction. Care has been taken to balance the mix of notation and words in mathematical statements.

Over 1500 problems are used to illustrate concepts, related to different topics, and introduce applications. In most examples, a question is first posed, then its solution is presented with appropriate details. The applications included in this text demonstrate the utility of Graph Theory, in the solution of real-world problem. This text includes applications to a wide-variety of areas, including computer science and engineering.

There are over 1000 exercises in the text with many different types of questions posed. There is an ample supply of straightforward exercises that develop basic skills, a large number of intermediate exercises and many challenging exercise sets. Problem sets are stated clearly and unambiguously, and all are carefully graded for various levels of difficulty.

It will be honest on my part to accept that it is not possible to include everything in one book.

Many people contributed directly or indirectly to the completion of this book. Thanks are due to my friends who were able to convince me that I should write this book.

I am grateful to my students, who always encouraged me and many times thanked me for writing this book.

Special thanks to my teachers, who made me realize that I can indeed write a book on "Graph Theory". Some pulled me down, some encouraged me and some gave me constructive suggestions. I am grateful to all of them.

I specially thank my parents, elder brothers, elder sister and maternal uncle, who tolerated me all along while I devoted my time to completing this book.

I experess my sincere thanks to the Chairman Mr. R.K. Gupta, the Managing Director Mr. Saumya Gupta, the Marketing Manager Mr. V.R. Babu and Mr. Vincent D. Souza, M/s New Age International (P) Ltd. Publishers, New Delhi, for their responsible work-done at every level in the publication of the book with high production standards.

Healthy criticism and suggestions to improve the quality and standards of the text are most welcome.

Bangalore C. Vasudev

This page intentionally left blank

This page intentionally left blank

This page intentionally left blank

CONTENTS

CHAPTER

1 Introduction to Graph Theory

INTRODUCTION

It is no coincidence that graph theory has been independently discovered many times, since it may quite properly be regarded as an area of **applied mathematics.**

The basic combinatorial nature of graph theory and a clue to its wide applicability are indicated in the words of Sylvester, **''The theory of ramification is one of pure colligation, for it takes no account of magnitude or position ; geometrical lines are used, but have no more real bearing on the matter than those employed in genealogical tables have in explaining the laws of procreation.''**

Indeed, the earliest recorded mention of the subject occurs in the works of Euler, and although the original problem he was considering might be regarded as a somewhat frivolous puzzle, it did arise from the physical world. Subsequent rediscoveries of graph theory by Kirchhoff and Cayley also had their roots in the physical world.

Kirchhoff's investigations of electric networks led to his development of the basic concepts and theorems concerning trees in graphs, while Cayley considered trees arising from the enumeration of organic chemical isomers. Another puzzle approach to graphs was proposed by Hamilton. After this, the celebrated four colour conjecture came into prominence and has been notorious ever since.

In the **present century,** there have already been a great many rediscoveries of graph theory which we can only mention most briefly in this chronological account.

Euler (1707–1782) became the father of graph theory as well as topology. Graph theory is considered to have begun in 1736 with the publication of Euler's solution of the Königsberg bridge problem. The graph theory is one of the few fields of mathematics with a definite birth date by ore.

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.

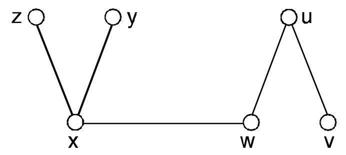
If $e = uv$ is an edge of a graph G, then we say that u and v are **adjacent** in G and that e joins u and v. (We may also say that each that of u and v is adjacent to or with the other).

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.



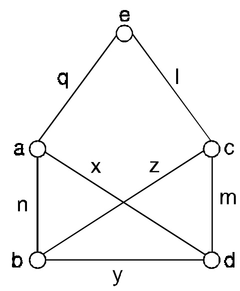
Every graph has a diagram associated with it. The vertex *u* and an edge *e* are **incident** with each other as are *v* and *e*. If two distinct edges say *e* and *f* are **incident** with a common vertex, then they are adjacent edges.

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 :

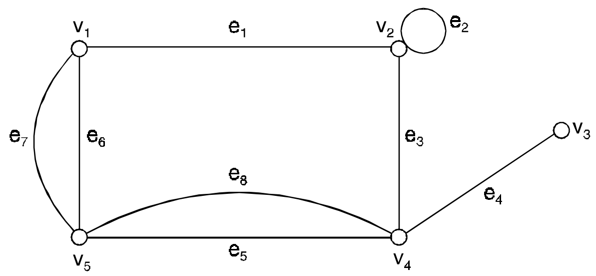
(1) Let V = {1, 2, 3, 4} and E = {{1, 2}, {1, 3}, {3, 2}. {4, 4}}.

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.

(3)



A graph with 5-vertices and 8-edges is called a **(5, 8) graph.**

1.2 DIRECTED AND UNDIRECTED GRAPHS

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.

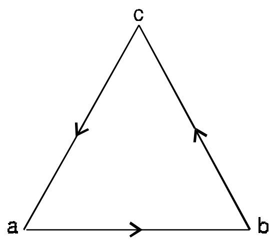


Fig. 1(*a*). Directed graph.

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.

1.2.2. Un-directed graph

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

In other words, if each edge of the graph G has no direction then the graph is called **un-directed graph.**

Figure 1(b) is an example of an undirected graph.

We can refer to an edge joining the vertex pair i and j as either (i, j) or (j, i).

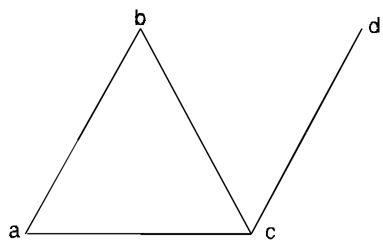


Figure 1(*b*). Un-directed graph.

1.3 BASIC TERMINOLOGIES

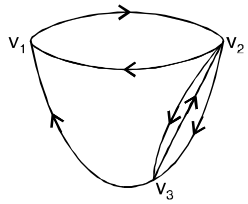
1.3.1 Loop : An edge of a graph that joins a node to itself is called **loop** or **self loop.**

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

1.3.2. Multigraph

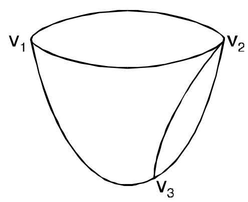
In a multigraph no loops are allowed but more than one edge can join two vertices, these edges are called **multiple edges** or parallel edges and a graph is called **multigraph.**

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

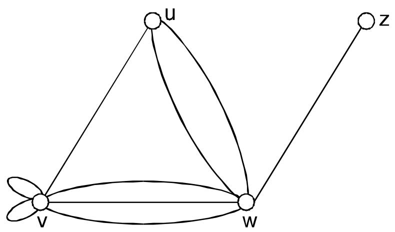
Fig. 2(*b*)

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.

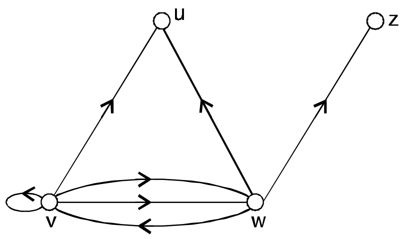
1.3.3. Pseudo graph

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



Un-directed Pseudo graph

Fig. 3(*a*)



Directed Pseudo graph

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 $\mathbf{d(v_i)}$.

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

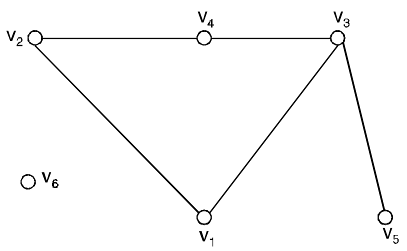


Fig. 4(*a*)

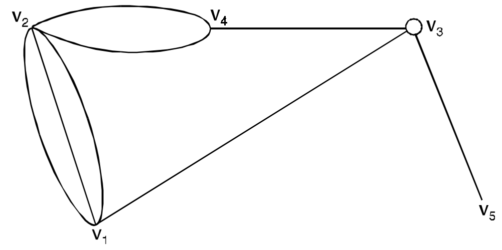


Fig. 4(*b*)

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 $\deg_G$ $\mathbf{(v_i)}$ or $\deg_G^{-1}$ $\mathbf{(v_i)}$**,** 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.

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

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

Proof : Since the degree of a vertex is the number of edges incident with that vertex, the sum of the degree counts the total number of times an edge is incident with a vertex.

Since every edge is incident with exactly two vertices, each edge gets counted twice, once at each end.

Thus the sum of the degrees equal twice the number of edges.

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

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

Proof : Let G = (V, E) a non directed graph.

Let U denote the set of even degree vertices in G and W denote the set of odd degree vertices.

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

$\Rightarrow \quad 2e - \sum_{v_i \in U} \deg_G (v_1) = \sum_{v_i \in W} \deg_G (v_1)$ ...(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 = $\{v_1, v_2, ... v_n\}$ 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.

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

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

Problem 1.1. *Show that, in any gathering of six people, there are either three people who all know each other or three people none of whom knows either of the other two (six people at a party).*

Solution. To solve this problem, we draw a graph in which we represent each person by a vertex and join two vertices by a solid edge if the corresponding people know each other, and by a dotted edge if not. We must show that there is always a solid triangle or a dotted triangle.

Let v be any vertex. Then there must be exactly five edges incident with v, either solid or dashed, and so at least three of these edges must be of the same type.

Let us assume that there are three solid edges (see figure 5) ; the case of atleast three dashed edges is similar.

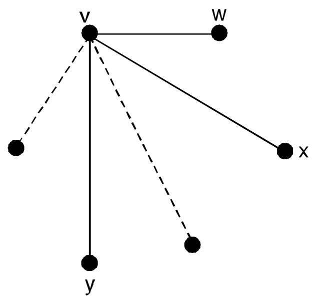


Fig. 5.

If the people corresponding to the vertices w and x know each other, then v, w and x form a solid triangle, as required.

Similarly, if the people corresponding to the vertices w and y, or to the vertices x and y, know each other, then we again obtain a solid triangle.

These three cases are shown in Figure (6).

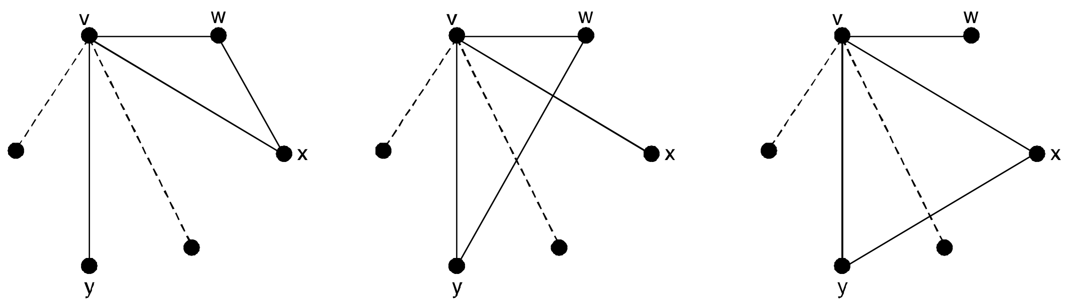


Fig. 6.

Finally, if no two of the people corresponding to the vertices *w*, *x* and *y* know each other, then *w*, *x* and *y* from a dotted triangle, as required (see figure (7).

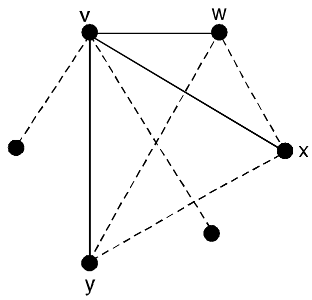


Fig. 7.

Problem 1.2. *Place the letters A, B, C, D, E, F, G, H into the eight circles in Figure (8), in such a way that no letter is adjacent to a letter that is next to it in the alphabet.*

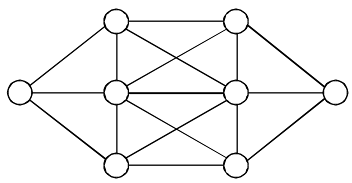


Fig. 8.

Solution. First note that trying all the possibilities is not a practical proposition, as there are 8 ! = 40320 ways of placing eight letters into eight circles.

Note that (*i*) the easiest letters to place are A and H, because each has only one letter to which it cannot be adjacent, namely, B and G, respectively.

(*ii*) the hardest circles to fill are those in the middle, as each is adjacent to six others.

This suggests that we place A and H in the middle circles. If we place A to the left of H, then the only possible positions for B and G are shown in Figure (9).

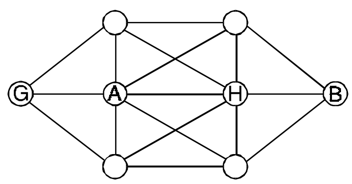


Fig. 9.

The letter C must now be placed on the left-hand side of the diagram, and F must be placed on the right-hand side.

It is then a simple matter to place the remaining letters, as shown in Figure (10).

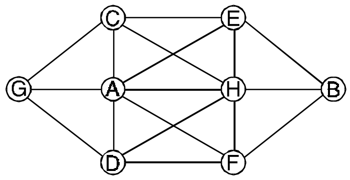


Fig. 10.

Problem 1.3. *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,

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

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

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).

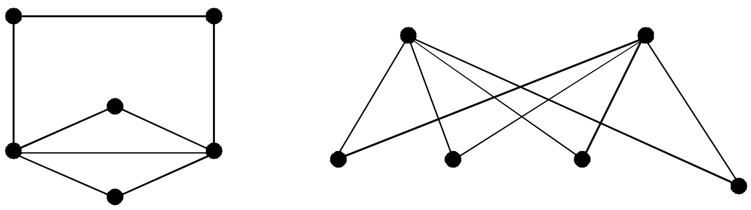


Fig. 11.

Problem 1.4. *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,

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

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

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

$\Rightarrow \quad 2P = 12 \quad \Rightarrow \quad$ P = 6 vertices are needed.

Problem 1.5. *It is possible to construct a graph with 12 vertices such that 2 of the vertices have degree 3 and the remaining vertices have degree 4.*

Solution. Suppose it is possible to construct a graph with 12 vertices out of which 2 of them are having degree 3 and remaining vertices are having degree 4.

Hence by handshaking lemma,

$$\sum_{i=1}^{12} d(v_i) = 2e \text{ where } e \text{ is the number of edges}$$

According to given conditions

$$(2 \times 3) + (10 \times 4) = 2e$$

$\Rightarrow$ $6 + 40 = 2e$

$\Rightarrow$ $2e = 46$ $\Rightarrow$ $e = 23$

It is possible to construct a graph with 23 edges and 12 vertices which satisfy given conditions.

Problem 1.6. *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 $\dfrac{P(P-1)}{2}$.

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

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

Hence such a graph does not exist.

Problem 1.7. *Show that the maximum degree of any vertex in a simple graph with P vertices is (P – 1).*

Solution. Let G be a simple graph with P-vertices. Consider any vertex v of G. Since the graph is simple (*i.e.,* without self loops and parallel edges), the vertex v can be adjacent to atmost remaining (P – 1) vertices.

Hence the maximum degree of any vertex in a simple graph with P vertices is (P – 1).

Problem 1.8. *Write down the vertex set and edge set of the following graphs shown in Figure 12(a) and 12(b).*

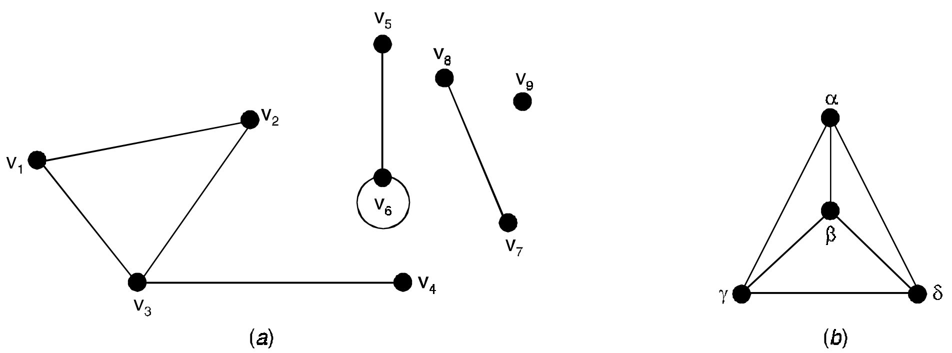


Fig. 12.

Solution. (*a*) V(G) = $\{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\}$

E(G) = $\{v_1v_2, v_1v_3, v_2v_3, v_3v_4, v_5v_6, v_6v_6, v_7v_8\}$

(*b*) V(G) = $\{\alpha, \beta, \gamma, \delta\}$

E(G) = $\{\alpha\beta, \alpha\gamma, \alpha\delta, \beta\delta, \beta\gamma, \gamma\delta\}$.

Problem 1.9. *Show that the size of a simple graph of order n cannot exceed* $^nC_2$.

Solution. Let G be a graph of order *n*.

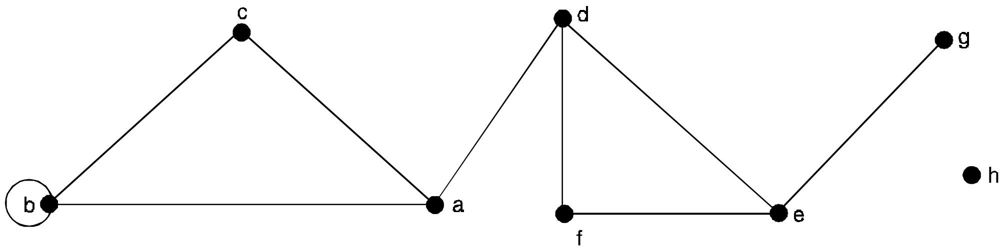
Let V be a vertex set of G.

Then cardinality of V is *n* and elements of E are distinct two elements subsets of V.

The number of ways we can choose two elements from a set V of *n* elements is $^nC_2$.

Thus, E may not have more than $^nC_2$ elements (edges).

Problem 1.10. *Find the degree sequence of the following graph.*



Solution. $\deg_G(a) = 3$, $\deg_G(b) = 4$, $\deg_G(c) = 2$

$\deg_G(d) = 3$, $\deg_G(e) = 3$, $\deg_G(f) = 2$

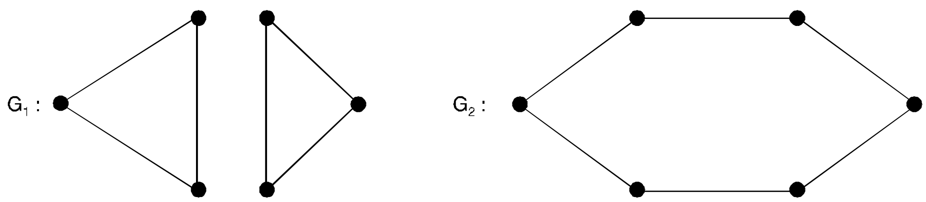
$\deg_G(g) = 1$, $\deg_G(h) = 0$.

Therefore, the degree sequence of the graph is 0, 1, 2, 2, 3, 3, 4.

Problem 1.11. *Construct two graphs having same degree sequence.*

Solution. The following two graphs have the same degree sequence.

The degree sequence of the graphs is 2, 2, 2, 2, 2, 2.



Problem 1.12. *Show that there exists no simple graph corresponds to the following degree sequence :*

(i) 0, 2, 2, 3, 4 *(ii) 1, 1, 2, 3* *(iii) 2, 2, 3, 4, 5, 5* *(iv) 2, 2, 4, 6.*

Solution. (*i*) to (*iii*) :

There are odd number of odd degree vertices in the graph.

Hence there exists no graph corresponds to this degree sequence.

(*iv*) Number of vertices in the graph is four and the maximum degree of a vertex is 6, which is not possible as the maximum degree cannot exceed one less than the number of vertices.

Problem 1.13. *Show that the total number of odd degree vertices of a (p, q)-graph is always even.*

Solution. Let v_1, v_2 ...... v_k be the odd degree vertices in G. Then, we have

$$\sum_{i=1}^{P} \deg_G (v_i) = 2q$$

i.e.,
$$\sum_{i=1}^{k} \deg_G (v_i) + \sum_{i=k+1}^{P} \deg_G (v_i) = 2q = \text{even number}$$

$$\Rightarrow \quad \sum_{i=1}^{k} \deg_G (v_i) = \text{even number} - \sum_{i=k+1}^{P} \deg_G (v_i)$$

$$\Rightarrow \quad \sum_{i=1}^{k} (\text{odd number}) = \text{even number} - \sum_{i=k+1}^{P} (\text{even number})$$

$$= \text{even number} - \text{even number}$$

$$= \text{even number.}$$

$\Rightarrow$ This implies that number of terms in the left-hand side of the equation is even.

Therefore, k is an even number.

Problem 1.14. *Show that the sequence 6, 6, 6, 6, 4, 3, 3, 0 is not graphical.*

Solution. To prove that the sequence is not graphical.

The given sequence is 6, 6, 6, 6, 4, 3, 3, 0

Resulting the sequence 5, 5, 5, 3, 2, 2, 0

Again consider the sequence 4, 4, 2, 1, 1, 0

Repeating the same 3, 1, 0, 0, 0

Since there exists no simple graph having one vertex of degree three and other vertex of degree one.

The last sequence is not graphical.

Hence the given sequence is also not graphical.

Problem 1.15. *Show that the following sequence is graphical. Also find a graph corresponding to the sequence 6, 5, 5, 4, 3, 3, 2, 2, 2.*

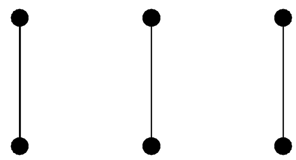
Solution. We can reduce the sequence as follows :

Given sequence	6, 5, 5, 4, 3, 3, 2, 2, 2
Reducing first 6 terms by 1 counting from second term	4, 4, 3, 2, 2, 1, 2, 2.
Writing in decreasing order	4, 4, 3, 2, 2, 2, 2, 1
Reducing first 4 terms by 1 counting from second	3, 2, 1, 1, 2, 2, 1
Writing in decending order	3, 2, 2, 2, 1, 1, 1
Reducing first 3 terms by 1, counting from second	1, 1, 1, 1, 1, 1

Sequence 1, 1, 1, 1, 1, 1 is graphical.

Hence the given sequence is also graphical.

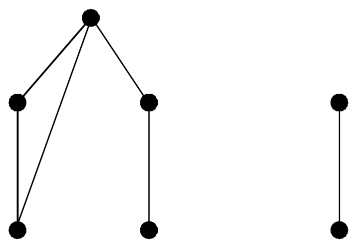
The graph corresponding to the sequence 1, 1, 1, 1, 1, 1 is given below



To obtain a graph corresponding to the given sequence, add a vertex to each of the vertices whose degrees are $t_1 - 1$, $t_2 - 1$, ... $t_s - 1$.

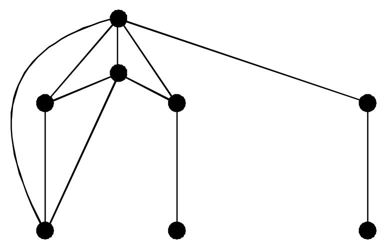
And repeat the process.

Step 1 :



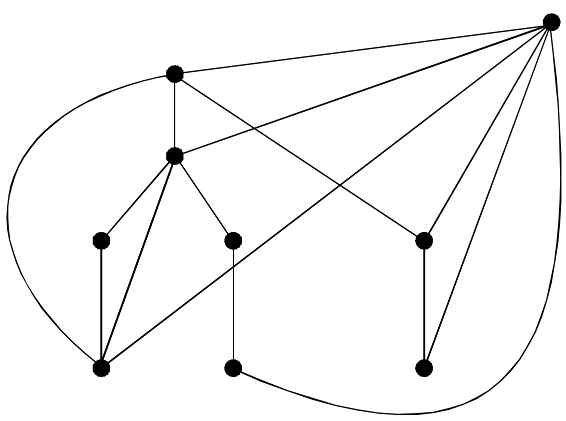
Degree sequence of this graph is 3, 2, 2, 2, 1, 1, 1

Step 2 :



Degree sequence of this graph is 4, 4, 3, 2, 2, 2, 2, 1.

Step 3 : Final graph



Degree sequence of this graph is 6, 5, 5, 4, 3, 3, 2, 2, 2.

Problem 1.16. *Show that no simple graph has all degrees of its vertices are distinct.*

(*i.e., in a degree sequence of a graph atleast one number should repeat.*)

Solution. Let G be a graph of order n.

Then there are n terms in the degree sequence of G. If no number (integer) in the degree sequence repeats, then only possible case it is of the form

$$0, 1, 2, 3, 4, ..., n - 1$$

Since maximum degree cannot exceed $n - 1$. But the last vertex of degree $n - 1$ should be adjacent to every other vertex of G, since G is simple.

Thus minimum degree of every vertex is one.

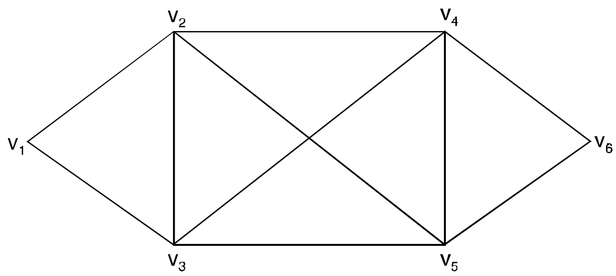
A contradiction to the fact that the degree of one vertex is zero.

Problem 1.17. *Is there a simple graph with degree sequence (1, 1, 3, 3, 3, 4, 6, 7) ?*

Solution. Assume there is such a graph. Then the vertex of degree 7 is adjacent to all other vertices, so in particular it must be adjacent to both vertices of degree 1.

Hence, the vertex v of degree 6 cannot be adjacent to either of the two vertices of degree 1.

Problem 1.18. *Find the degree of each vertex of the following graph :*

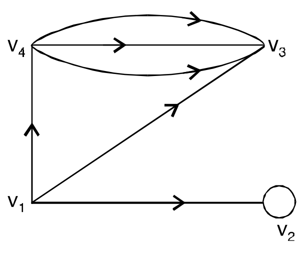


Solution. It is an undirected graph. Then

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

deg (v_4) = 4, deg (v_5) = 4, deg (v_6) = 2.

Problem 1.19. *Find the in degree out degree and of total degree of each vertex of the following graph.*



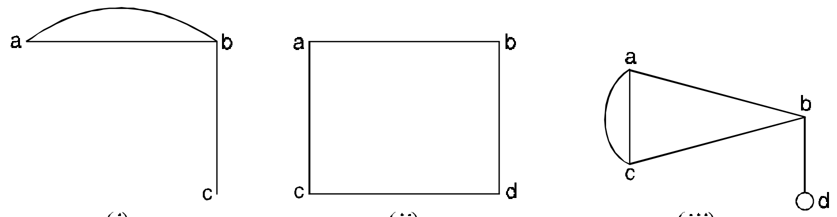
Solution. It is a directed graph

in deg (v_1) = 0, out deg (v_1) = 3, total deg (v_1) = 4

in deg (v_2) = 2, out get (v_2) = 1, total deg (v_2) = 3 in deg (v_3) = 4, out deg (v_3) = 0, total deg (v_3) = 4

in deg (v_4) = 1, out deg (v_4) = 3, total deg (v_4) = 4.

Problem 1.20. *State which of the following graphs are simple ?*



(*i*) (*ii*) (*iii*)

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

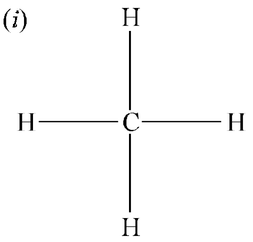
(*ii*) The graph is a simple graph, it does not contain loop and parallel edge.

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

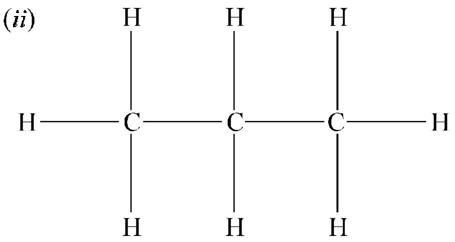
Problem 1.21. *Draw the graphs of the chemical molecules of*

(i) Methane (CH_4) *(ii) Propane (C_3H_8).*

Solution. (*i*)



(*ii*)



Problem 1.22. *Show that the degree of a vertex of a simple graph G on n vertices cannot exceed n – 1.*

Solution. Let *v* be a vertex of G, since G is simple, no multiple edges or loops are allowed in G.

Thus *v* can be adjacent to atmost all the remaining *n* – 1 vertices of G.

Hence *v* may have maximum degree *n* – 1 in G.

i.e., $$0 \le \deg_G (v) \le n - 1 \text{ for all } v \in V(G).$$

Problem 1.23. *Does there exists a simple graph with seven vertices having degrees (1, 3, 3, 4, 5, 6, 6) ?*

Solution. Suppose there exists a graph with seven vertices satisfying the given properties.

Since two vertices have degree 6, each of these two vertices is adjacent with every other vertex.

Hence the degree of each vertex is at least 2, so that the graph has no vertex of degree 1, which is a contradiction.

Hence there does not exist a simple graph with the given properties.

Problem 1.24. *Is there a simple graph corresponding to the following degree sequences ?*

(i) (1, 1, 2, 3) *(ii) (2, 2, 4, 6).*

Solution. (*i*) There are odd number (3) of odd degree vertices, 1, 1 and 3.

Hence there exist no graph corresponding to this degree sequence.

(*ii*) Number of vertices in the graph sequence is 4, and the maximum degree of a vertex is 6, which is not possible as the maximum degree cannot exist on less than the number of vertices.

Problem 1.25. *Show that the maximum number of edges in a simple graph with n vertices is* $\frac{n(n-1)}{2}$.

Solution. By the handshaking theorem,

$$\sum_{i=1}^{n} d(v_i) = 2e$$

where e is the number of edges with n vertices in the graph G.

$\Rightarrow \quad d(v_1) + d(v_2) + \ldots\ldots + d(v_n) = 2e \qquad \ldots(1)$

Since we know that the maximum degree of each vertex in the graph G can be $(n - 1)$.

Therefore, equation (1) reduces

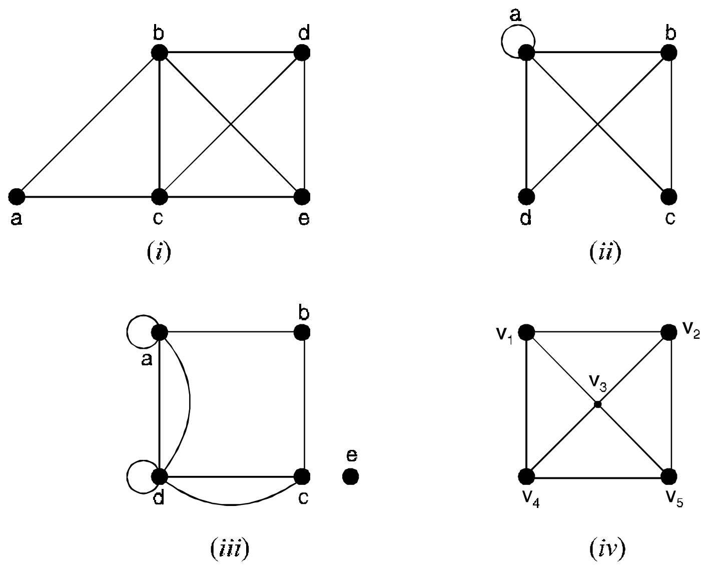
$$(n-1) + (n-1) + \ldots\ldots \text{ to } n \text{ terms} = 2e$$

$\Rightarrow \quad n(n-1) = 2e$

$$\Rightarrow \qquad e = \frac{n(n-1)}{2}.$$

Hence the maximum number of edges in any simple graph with n vertices is $\frac{n(n-1)}{2}$.

Problem 1.26. *Consider the following graphs and determine the degree of each vertex :*

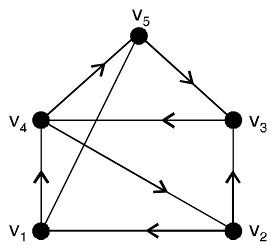


Solution. (*i*) deg (a) = 2, deg (b) = 4, deg (c) = 4, deg (d) = 3, deg (e) = 3 (*ii*) deg (a) = 5, deg (b) = 2, deg (c) = 3, deg (d) = 6, deg (e) = 0

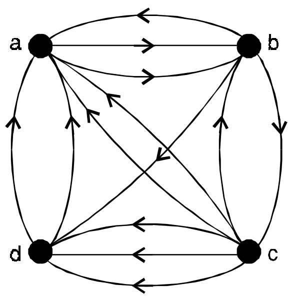
(*iii*) deg (a) = 5, deg (b) = 3, deg (c) = 2, deg (d) = 2,

(*iv*) Every vertex has degree 4.

Problem 1.27. *Find the in-degree and out-degree of each vertex of the following directed graphs :*



(*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.

Problem 1.28. *Draw a graph having the given properties or explain why no such graph exists.*

(i) Graph with four vertices of degree 1, 1, 2 and 3.

(ii) Graph with four vertices of degree 1, 1, 3 and 3

(iii) Simple graph with four vertices of degree 1, 1, 3 and 3

(iv) Graph with six vertices each of degree 3

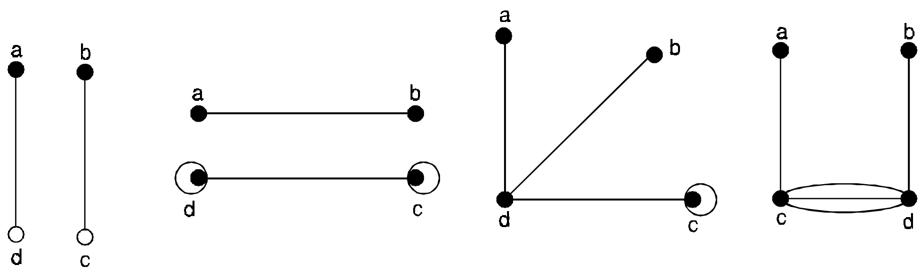
(v) Graph with six vertices and four edges

(vi) Graph with five vertices of degree 3, 3, 3, 3, 2

(vii) Graph with five vertices of degree 0, 1, 2, 2, 3.

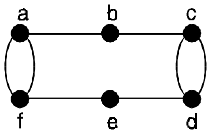
Solution. (*i*) No such graphs exists, total degree is odd.

(*ii*)

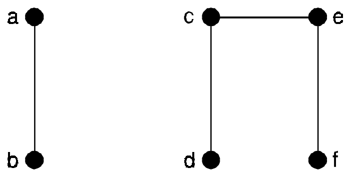


(*iii*) No simple graph.

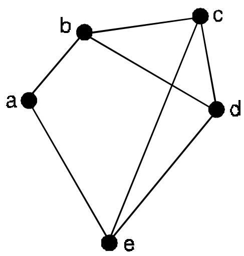
(*iv*)



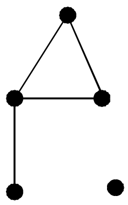
(*v*)



(*vi*)



(*vii*)

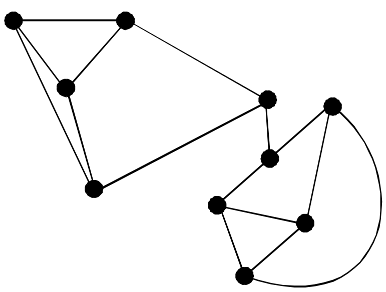


Problem 1.29. *If the simple graph G has V vertices and e edges, how many edges does G′ (complement of G) have ?*

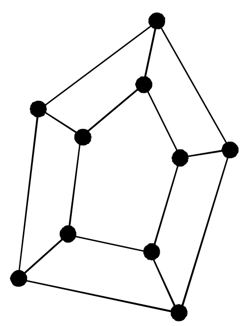
Solution. $\dfrac{v(v-1)}{2-e}$.

Problem 1.30. *Construct a 3-regular graph on 10 vertices.*

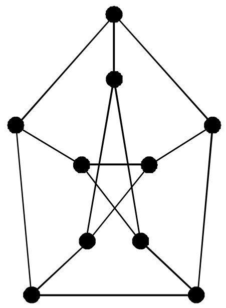
Solution. The following graphs are some examples of 3-regular graphs on 10 vertices.



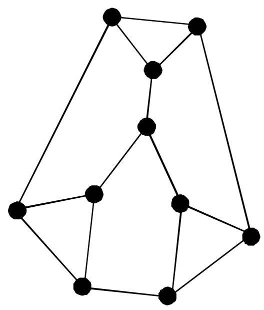
(*i*)



(*ii*)



(*iii*)

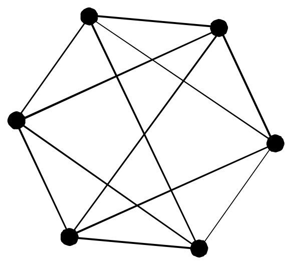


(*iv*)

Problem 1.31. *Does there exists a 4-regular graph on 6 vertices ? If so construct a graph.*

Solution. We have $q = \dfrac{P \times r}{2} = \dfrac{6 \times 4}{2} = 12$

Hence 4-regular graph on 6-vertices is possible and it contains 12 edges. One of the graph is shown below.



Every 4-regular graph contains a 3-regular graph.

Problem 1.32. *What is the size of an r-regular (p, q)-graph.*

Solution. Since G is an r-regular graph.

By the definition of regularity of G.

We have $\deg_G(v_i) = r$ for all $v_i \in V(G)$

But $2q = \sum_{i=1}^{P} \deg_G(v_i)$

$$2q = \sum_{i=1}^{P} r = P \times r$$

$$\Rightarrow \quad q = \frac{P \times r}{2}.$$

Problem 1.33. *Does a 3-regular graph on 14 vertices exist ? What can you say on 17 vertices ?*

Solution. We have $q = \dfrac{P \times r}{2}$

given $r = 3$, $P = 14$

Now $q = \dfrac{14 \times 3}{2} = 21$, is a positive integer.

Hence 3-regular graphs on 14 vertices exist.

Further, if P = 17, then $q = \dfrac{P \times r}{2} = \dfrac{17 \times 3}{2} = \dfrac{51}{2}$ is not a positive integer.

Hence no 3-regular graphs on 17 vertices 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.

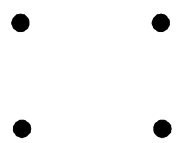


Fig. 13.

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 $\dfrac{n(n-1)}{2}$ edges.

The graphs K_n for $n = 1, 2, 3, 4, 5, 6$ are show in Figure 14.

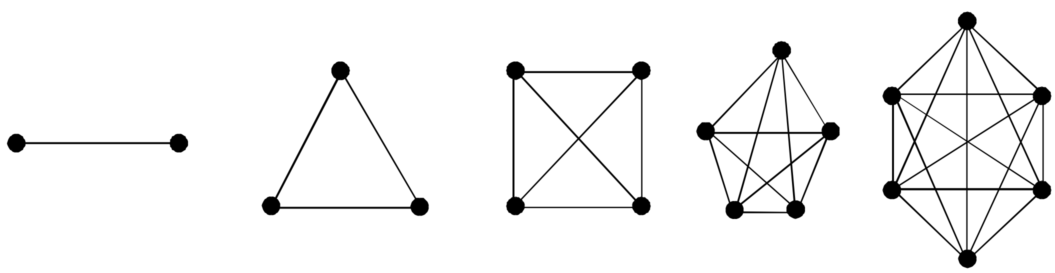


Fig. 14.

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.

1.7.4. Cycles

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

The cyles c_3, c_4, c_5 and c_6 are shown in Figure 15.

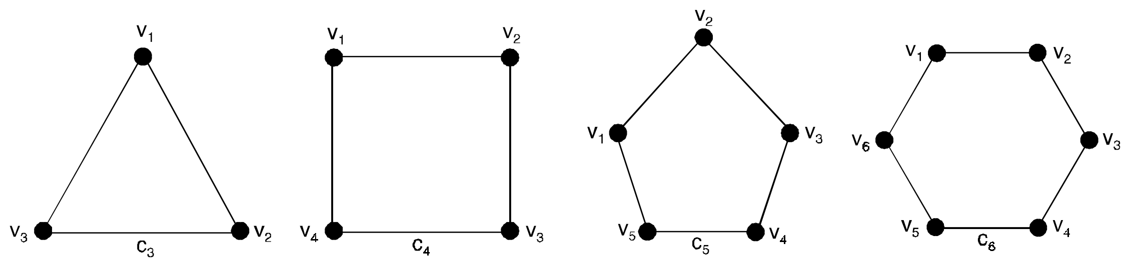


Fig. 15. Cycles C_3, C_4, C_5 and C_6.

1.7.5. Wheels

The wheel W_n is obtained when an additional vertex to the cycle c_n, for $n \geq 3$, and connect this new vertex to each of the n vertices in c_n, by new edges. The wheels W_3, W_4, W_5 and W_6 are displayed in Figure 16.

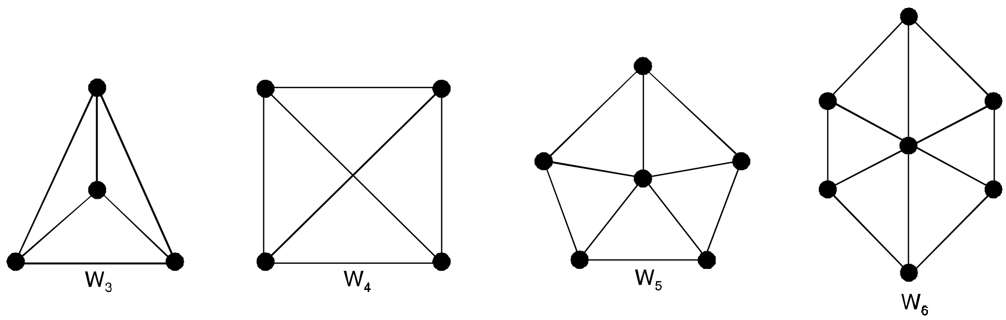


Figure 16. The wheels W_3, W_4, W_5 and W_6

1.7.6. Platonic graph

The graph formed by the vertices and edges of the five regular (platonic) solids—The tetrahedron, octahedron, cube, dodecahedron and icosahedron.

The graphs are shown in Figure 17.

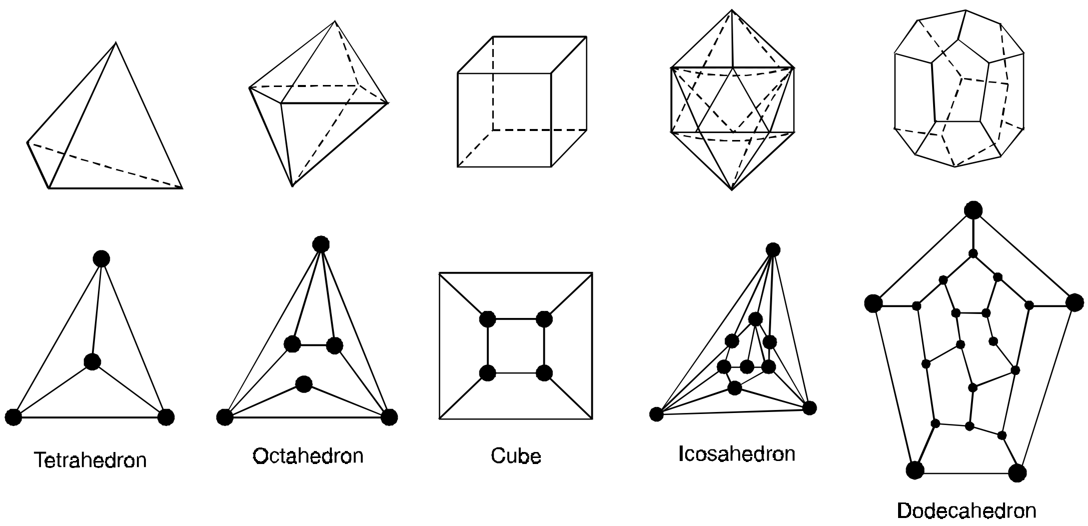


Fig. 17.

1.7.7. N-cube

The N-cube denoted by Q_n, is the graph that has vertices representing the 2^n bit strings of length *n*. Two vertices are adjacent if and only if the bit strings that they represent differ in exactly one bit position. The graphs Q_1, Q_2, Q_3 are displayed in Figure 18. Thus Q_n has 2^n vertices and $n \,.\, 2^{n-1}$ edges, and is regular of degree *n*.

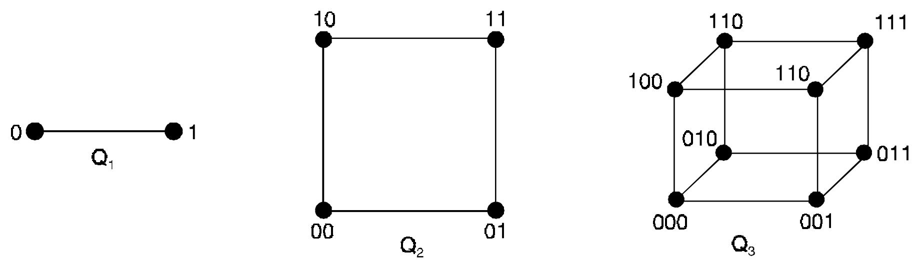
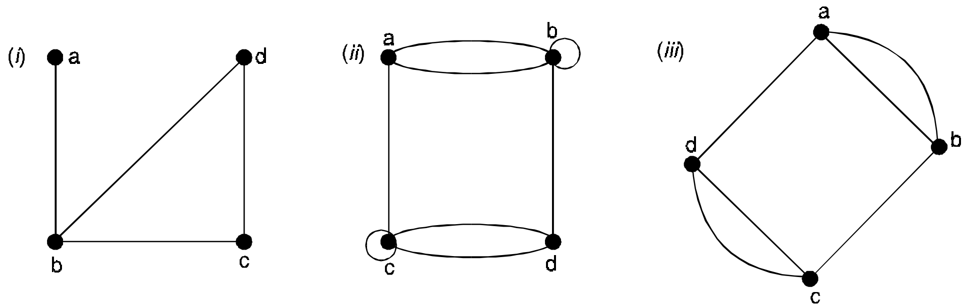


Fig. 18. The *n*-cube Q_n for $n = 1, 2, 3$.

Problem 1.34. *Determine whether the graphs shown is a simple graph, a multigraph, a pseudograph.*



Solution. (*i*) Simple graph

(*ii*) Pseudograph

(*iii*) Multigraph.

Problem 1.35. *Consider the following directed graph G : V(G) = {a, b, c, d, e, f, g}*

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

(*i*) *Identify any loops or parallel edges.*

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

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

(*iv*) *Find the subgraph H of G determined by the vertex set V′ = {a, b, c, d}.*

Solution. (*i*) (*a*, *a*) and (*g*, *g*) are loops

(*a*, *a*) and (*a*, *e*) are parallel edges.

(*ii*) No sources

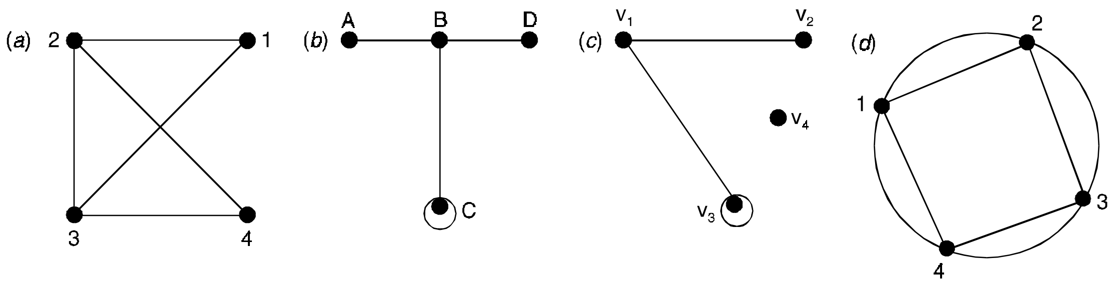
(*iii*) No sinks

(*iv*) V′ = {*a*, *b*, *c*, *d*}

E′ = {(*a*, *a*), (*d*, *b*)}

H = H(V′, E′).

Problem 1.36. *Consider the following graphs, determine the (i) vertex set and (ii) edge set.*



Solution. (*a*) (*i*) Vertex set V = {1, 2, 3, 4},

(*ii*) Edge set E = {(1, 2), (1, 3), (2, 3), (2, 4), (3, 4)}

(*b*) (*i*) Vertex set V = {A, B, C, D}

(*ii*) Edge set E = {(A, B), (B, C), (B, D), (C, C)} (*c*) (*i*) Vertex set V = $\{v_1, v_2, v_3, v_4\}$

(*ii*) Edge set E = $\{(v_1, v_2), (v_1, v_3), (v_3, v_3)\}$

(*d*) (*i*) Vertex set V = {1, 2, 3, 4}

(*ii*) Edge set E = {(1, 2), (2, 3), (3, 4), (4, 1)}

All edges are double edges.

Problem 1.37. *How many vertices and how many edges do the following graphs have ?*

(*i*) K_n (*ii*) C_n (*iii*) W_n (*iv*) $K_{m,\,n}$ (*v*) Q_n.

Solution. (*i*) n vertices and $\dfrac{n(n-1)}{2}$ edges.

(*ii*) n vertices and n edges

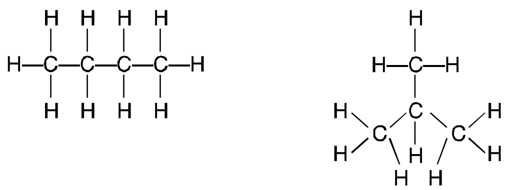
(*iii*) $n + 1$ vertices and $2n$ edges

(*iv*) $m + n$ vertices and mn edges

(*v*) 2^n vertices and $n \,.\, 2^{n-1}$ edges.

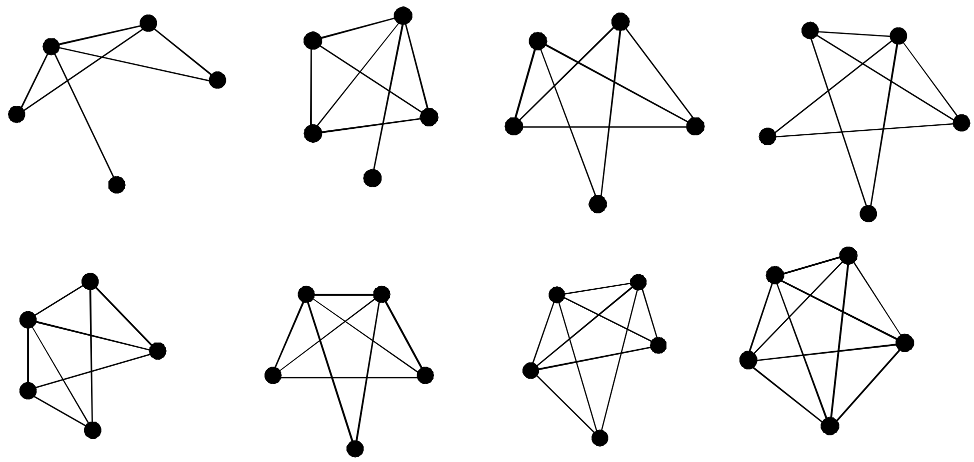
Problem 1.38. *There are two different chemical molecules with formula C_4H_{10} (isobutane). Draw the graphs corresponding to these molecules.*

Solution.



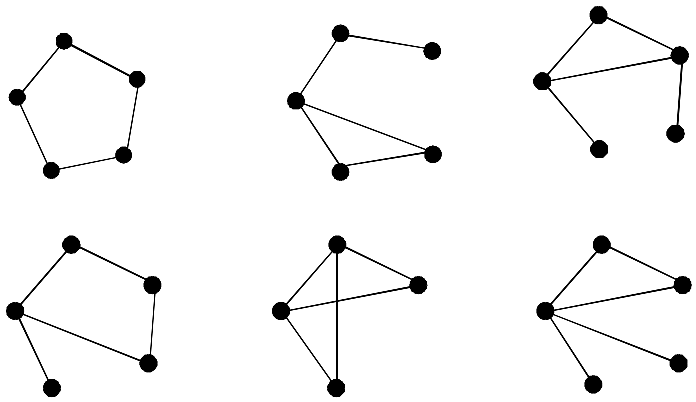
Problem 1.39. *Draw all eight graphs with five vertices and seven or more edges.*

Solution.



Problem 1.40. *Draw all six graphs with five vertices and five edges.*

Solution.



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.

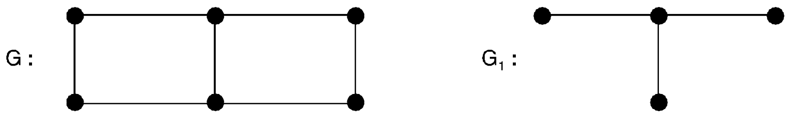


Fig. 19. G_1 is a subgraph of G.

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.

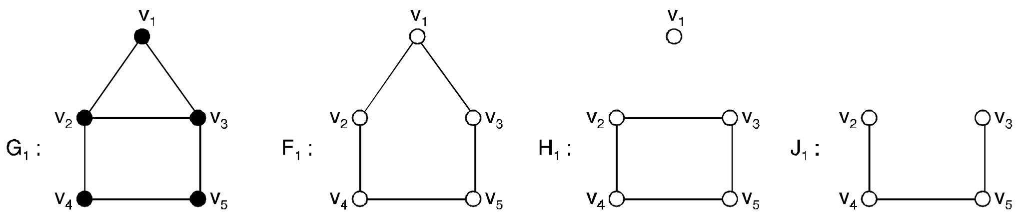


Fig. 20.

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.

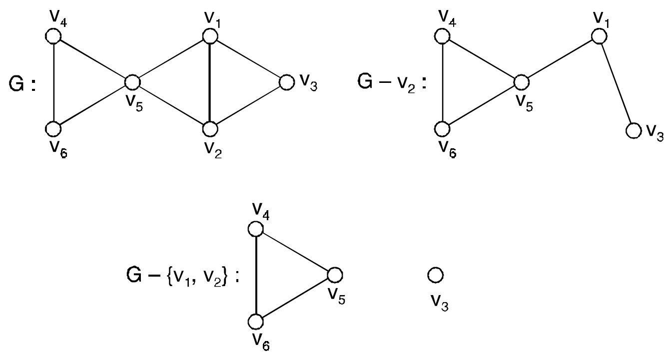


Fig. 21(*a*). Deleting vertices from a graph.

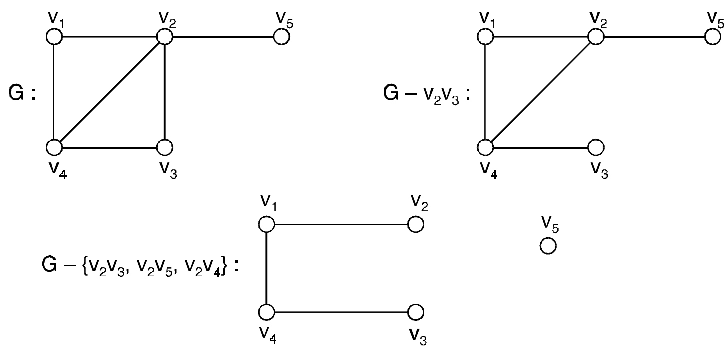
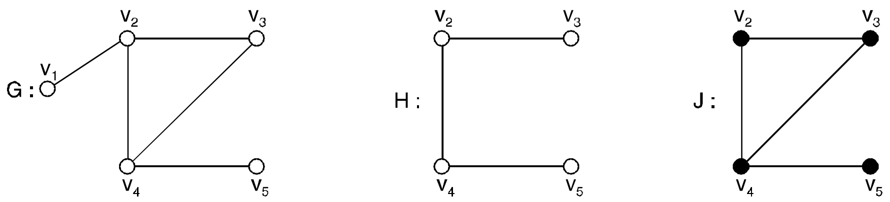


Fig. 21(*b*). Deleting edges from a graph.

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 GRAPHS ISOMORPHISM

Let $G_1 = (v_1, E_1)$ and $G_2 = (v_2, E_2)$ be two graphs. A function $f : v_1 \to v_1$ 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.

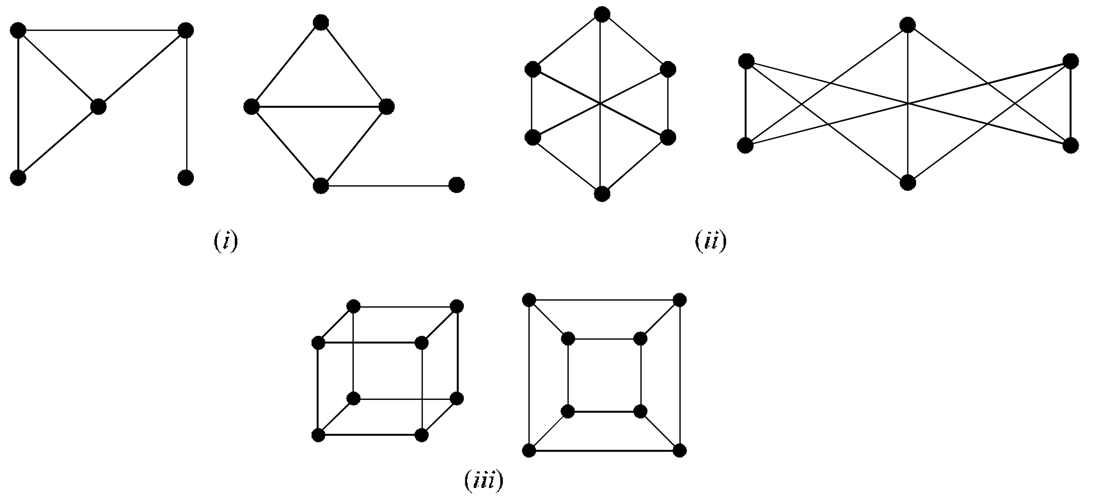


Fig. 22(*i*), (*ii*) (*iii*) Isomorphic pair of graphs

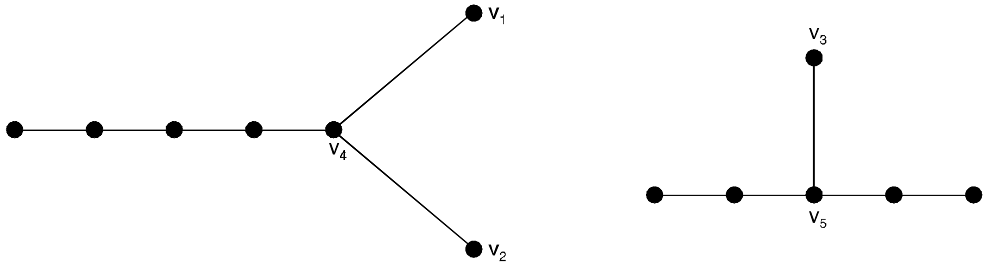
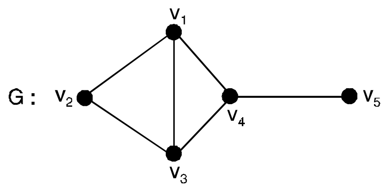
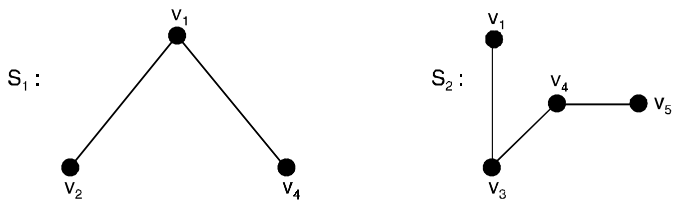


Fig. 23. Two graphs that are not isomorphic.

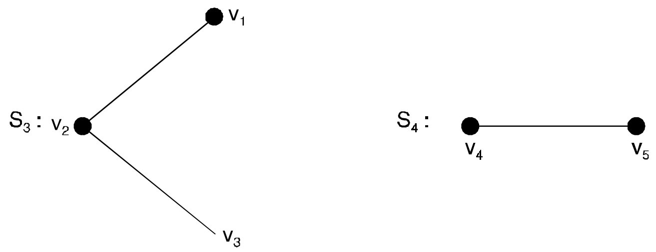
Problem 1.41. *Construct two edge-disjoint subgraphs and two vertex disjoint subgraphs of a graph G shown below*



Solution.



The graphs S_1 and S_2 are edge-disjoint subgraphs of G.



S_3 and S_4 are vertex disjoint subgraphs of G which are also edge-disjoint subgraphs of G.

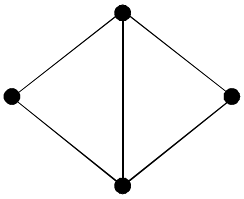
Problem 1.42. *Does there exist a proper subgraph S of G whose size is equal to the size of the graph ?*

Solution. Yes, consider the graph G shown in Figure below.

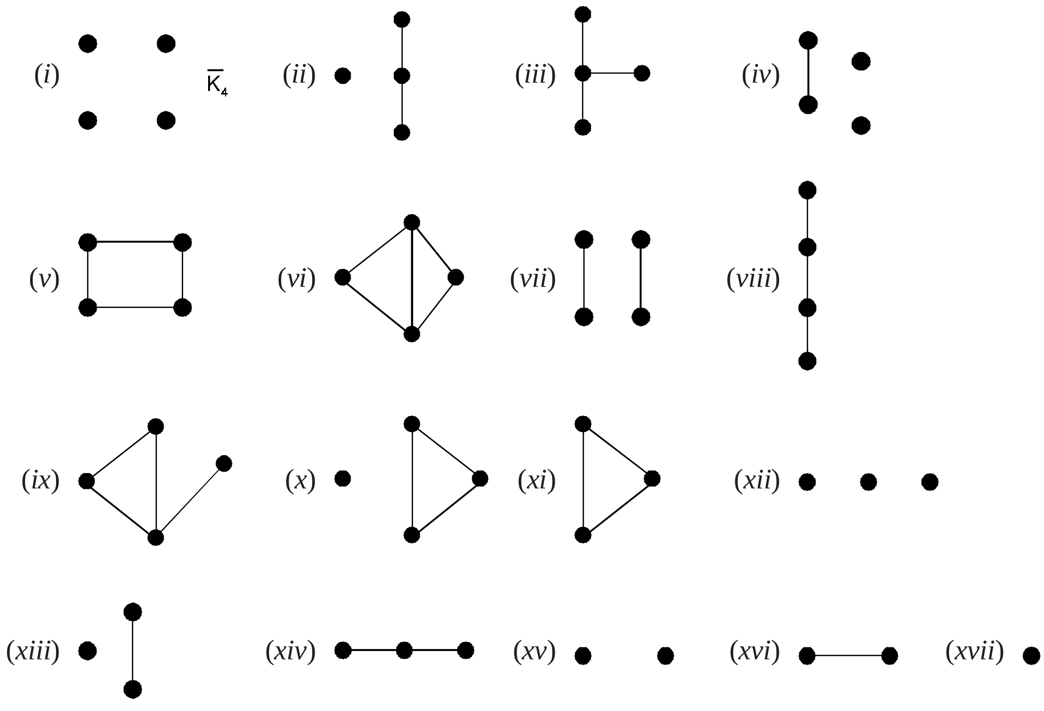
The graph S is a subgraph of G with $V(S) \subset E(G)$ and $E(S) = E(G)$.

The graph G Subgraph S

Problem 1.43. *Write down all possible non-isomorphic subgraphs of the following graphs G. How many of they are spanning subgraphs ?*

G : 

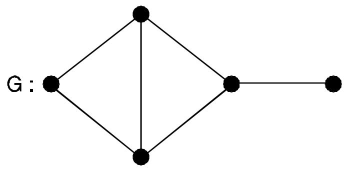
Solution. Its possible all (non-isomorphic) subgraphs are



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

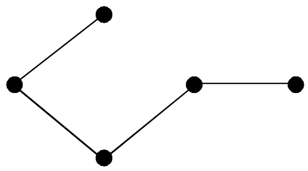
All the graphs except (*vi*) are proper subgraphs of G.

Problem 1.44. *Construct three non-isomorphic spanning subgraphs of the graph G shown below :*

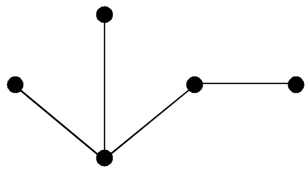


Solution. Three non-isomorphic subgraphs are

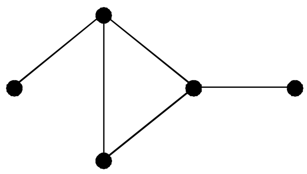
(*i*)



(*ii*)



(*iii*)



Problem 1.45. *Find the total number of subgraphs and spanning subgraphs in K_6, L_5 and Q_3.*

Solution. In graph K_6, we have | V | = 6 and | E | = 15

Thus, total number of subgraph is

$$(2^6 - 1) \times 2^{15} = 63 \times 32768 = 2064384$$

The total number of spanning subgraph is : $2^{15} = 32768$.

In the linear graph L_5, we have | V | = 5 and | E | = 4

Thus, total number of subgraph is

$$(2^5 - 1) \times 2^4 = 31 \times 16 = 496.$$

The total number of spanning subgraph is : $2^4 = 16$.

In the 3-cube graph Q_3, we have | V | = 8 and | E | = 12

Thus, total number of subgraph is

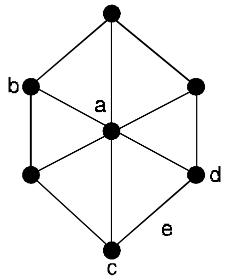
$$(2^8 - 1) \times 2^{12} = 127 \times 4096 = 520192$$

The total number of spanning subgraphs is

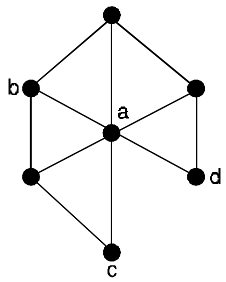
$$2^{12} = 4096.$$

Problem 1.46. *For the graph G shown below, draw the subgraphs*

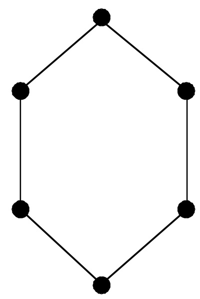
(*i*) *G – e* (*ii*) *G – a* (*iii*) *G – b*.



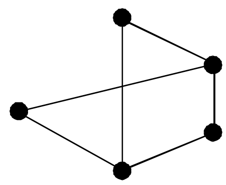
Solution. (*i*) After deleting the edge $e = (c, d)$ from the graph G, we get a subgraph G – e as shown below



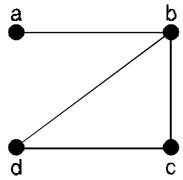
(*ii*) After deleting the vertex a from the graph G, and all edges incident on this vertex, we set the subgraph G – a as shown below :



(*iii*) The subgraph is obtained after deleting the vertex *b*.



Problem 1.47. *Consider the graph G(V, E) shown below, determine whether or not* $H(V_1, E_1)$ *is a subgraph of G, where*



(*i*) $V_1 = \{a, b, d\}$ $E_1 = \{(a, b), (a, d)\}$

(*ii*) $V_1 = \{a, b, c, d\}$ $E_1 = \{(b, c), (b, d)\}$

Solution. (*i*) H is not a subgraph because (a, d) is not an edge in G.

(*ii*) H is a subgraph because it satisfies condition for a subgraph of the given graph G.

Problem 1.48. *Find all possible non-isomorphic induced subgraphs of the following graph G corresponding to the three element subsets of the vertex set of G*

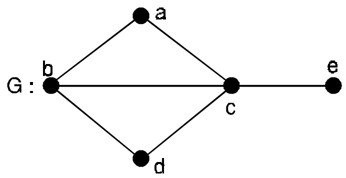
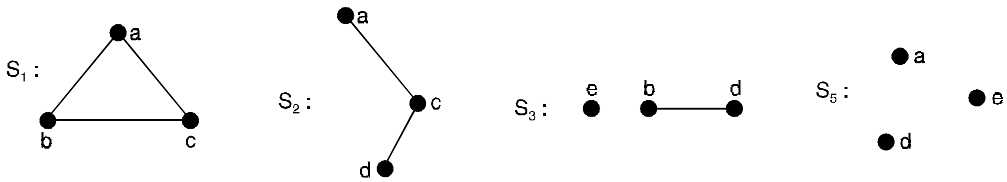


Fig. 24.

Solution.



The subgraph S shown in Figure (25) of the above graph G shown in Figure 24 is not a induced subgraph of G.

For the edge (a, d) of G can be added to S. The graph obtained by adding this edge is again a subgraph of

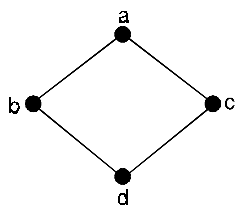
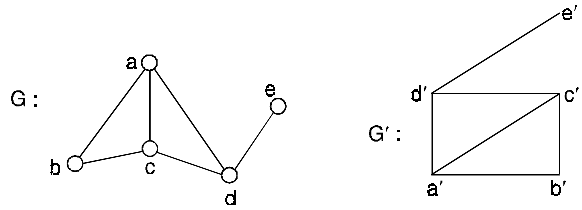


Fig. 25.

Note : The graph G is itself a maximal subgraph of G.

Problem 1.49. *Show that the following graphs are isomorphic*



Solution. Let $f: G \to G'$ be any function defined between two graphs degrees of the graph G and G′ are as follows :

deg (G)	deg (G′)
deg (a) = 3	deg (a') = 3
deg (b) = 2	deg (b') = 2
deg (c) = 3	deg (c') = 3
deg (d) = 3	deg (d') = 3
deg (e) = 1	deg (e') = 1

Each has 5-vertices and 6-edges.

$d(a) = d(a') = 3$

$d(b) = d(b') = 2$

$d(c) = d(c') = 3$

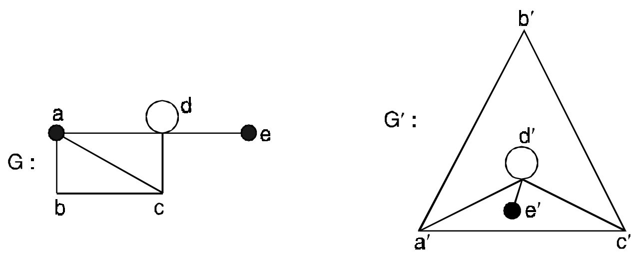
$d(d) = d(d') = 3$

$d(e) = d(e') = 1$

Hence the correspondence is $a - a'$, $b - b'$, ..., $e - e'$.

Therefore, the given two graphs are isomorphic.

Problem 1.50. *Show that the following graphs are isomorphic.*



Solution. Let $f: G \to G'$ be any function defined between two graphs degrees of the graphs G and G′ are as follows :

deg (G)	deg (G′)
deg (a) = 3	deg (a') = 3
deg (b) = 2	deg (b') = 2
deg (c) = 3	deg (c') = 3

deg (d) = 5 deg (d') = 5

deg (e) = 1 deg (e') = 1

Each has 5-vertices, 6-edges and 1-circuit.

deg(a) = deg(a') = 3

deg(b) = deg(b') = 2

deg(c) = deg(c$'$) = 3

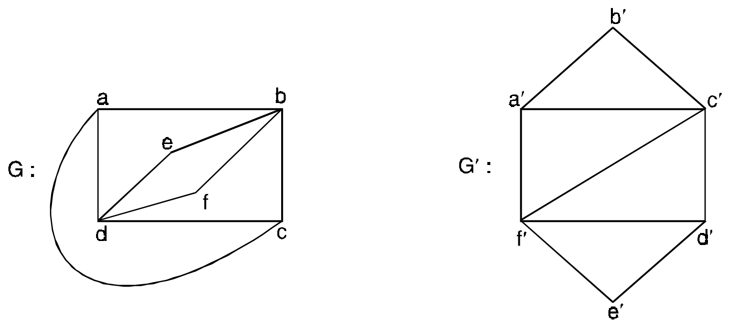
deg(d) = deg(d') = 5

deg(e) = deg(e') = 1

Hence the correspondence is $a - a'$, $b - b'$, ..., $e - e'$.

Therefore, the given two graphs G and G$'$ are isomorphic.

Problem 1.51. *Are the 2-graphs, is given below, is isomorphic ? Give a reason.*



Solution. Let us enumerate the degree of the vertices

Vertices of degree 4 : $b - f'$

$d - c'$

Vertices of degree 3 : $a - a'$

$c - d'$

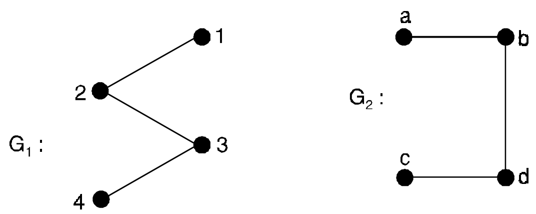
Vertices of degree 2 : $e - b'$

$f - e'$

Now the vertices of degree 3, in G are a and c and they are adjacent in G$'$, while these are a' and d' which are not adjacent in G$'$.

Hence the 2-graphs are not isomorphic.

Problem 1.52. *Show that the two graphs shown in Figure are isomorphic.*



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

$E(G_1) = \{\{1, 2\}, \{2, 3\}, \{3, 4\}\}$ and $E(G_2) = \{\{a, b\}, \{b, d\}, \{d, c\}\}$

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

$$f(1) = a,\ f(2) = b,\ f(3) = d, \text{ and } f(4) = c$$

f is clearly one-one and onto, hence an isomorphism.

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

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

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

and $\{1, 3\} \notin E(G_1)$ and $\{f(1), f(3)\} = \{a, d\} \notin E(G_2)$

$\{1, 4\} \notin E(G_1)$ and $\{f(1), f(4)\} = \{a, c\} \notin E(G_2)$

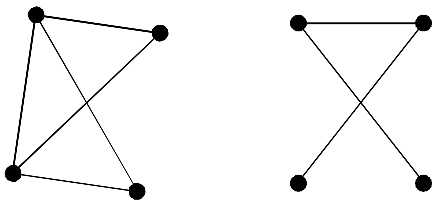
$\{2, 4\} \notin E(G_1)$ and $\{f(2), f(4)\} = \{b, c\} \notin E(G_2)$.

Hence f preserves adjacency as well as non-adjacency of the vertices.

Therefore, G_1 and G_2 are isomorphic.

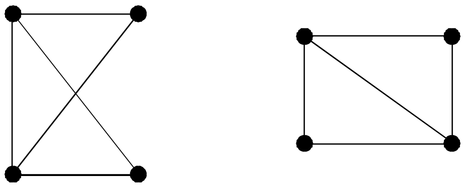
Problem 1.53. *For each pair of graphs shown, either label the graphs so as to exhibit an isomorphism or explain why the graphs are not isomorphic.*

(*i*)



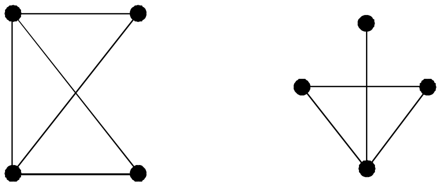
(a) (b)

(*ii*)



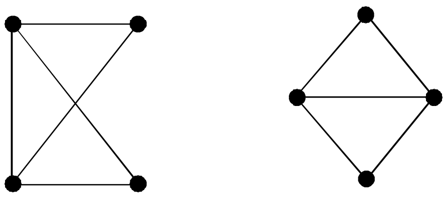
(a) (b)

(*iii*)



(a) (b)

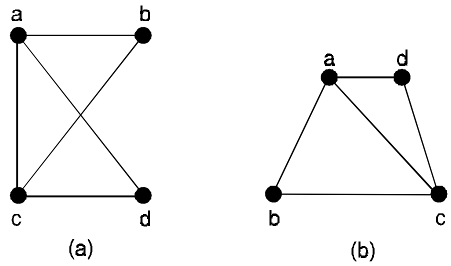
(*iv*)



(a) (b)

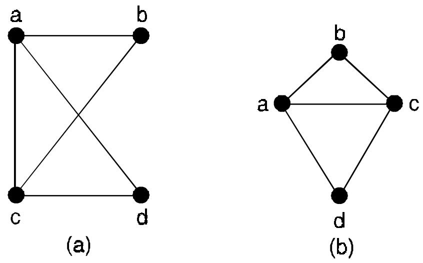
Solution. (*i*) The graphs are not isomorphic because (*a*) has 5-edges and (*b*) has 4-edges.

(*ii*) The graphs are isomorphic, as shown by the labelling



(*iii*) The graphs are not isomorphic because (*b*) has a vertex of degree 1 and (*a*) does not have.

(*iv*) The graphs are isomorphic, as shown by the labelling



Problem 1.54. *Whether the following pair of non-directed graphs in figure (26) are isomorphic or not ? Justify your answer ?*

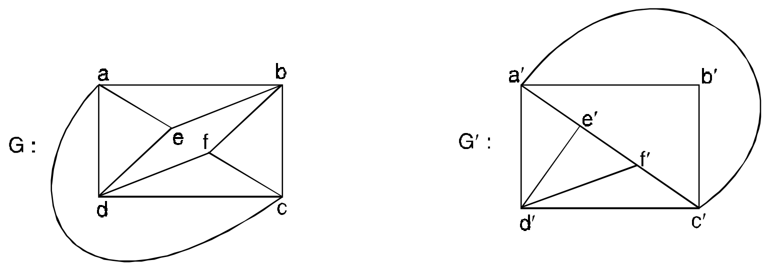


Fig. 26.

Solution. Here, G′ has vertex *b*′ of degree 2, while G has no vertex of degree 2.

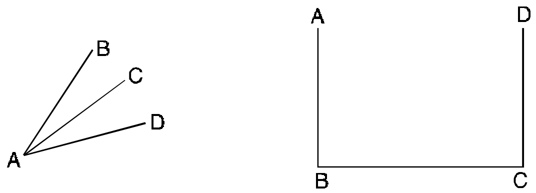
Hence, they are not isomorphic.

Problem 1.55. *How many different non-isomorphic trees are possible for a graph of order 4 ? Draw all of them.*

Solution. The sum of the degrees of the 4-vertices equals

$$2(e) = 2(n - 1) = 2n - 2 = 8 - 2 = 6$$

Hence, the degree of 4-vertices are (2, 2, 1, 1) or (3, 1, 1, 1), they are drawn as shown in Figure below



Problem 1.56. *Draw a cycle graph which is isomorphic to its complement.*

Solution. First we draw G and the complement of G denoted G′, by drawing edges between vertices which are non-adjacent in G.

The vertices in G′ are labelled so as to corresponds to those of G as follows :

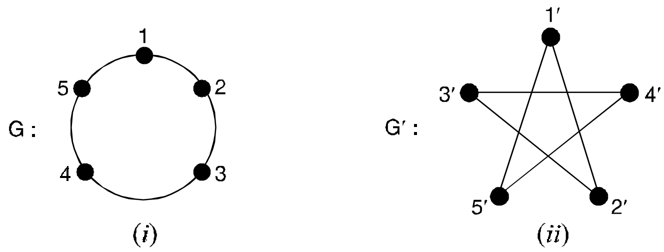


Fig. 27.

From Figure (27)

Vertices in G	*Vertices in G′*
1	1′
2	2′
3	3′
4	4′
5	5′

This labelling ensures that 5′ and 2′ are adjacent to 1′ in G′, while 5 and 2 are adjacent to 1 in G, 3′ and 1′ are adjacent to 2′ in G′, while 3 and 1 are adjacent to 2 in G.

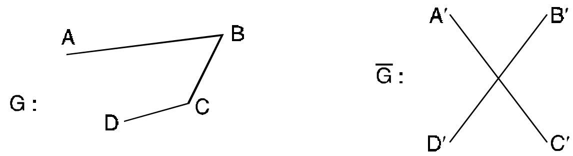
Also $d(i') = d(i)$ for all i.

Hence G and G′ are isomorphic.

Problem 1.57. *If a simple graph with n-vertices is isomorphic with its complement, how many vertices will that have ? Draw the corresponding graph.*

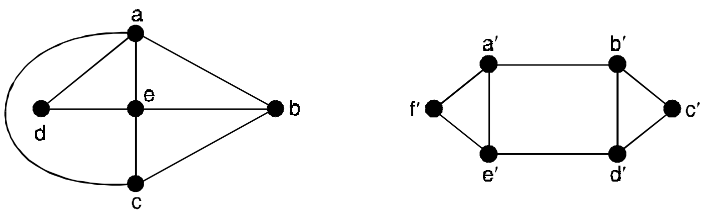
Solution. If e is the number of edges of G and $\bar{e}$ the number of edges in the complement $\overline{G}$, then

$e = \bar{e} = \dfrac{n(n+1)}{4}$. Hence n or $n + 1$ must be divisible by 4.

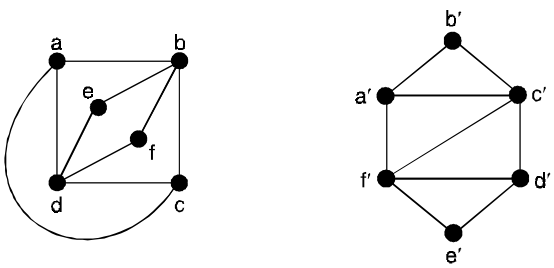


Problem 1.58. *Determine whether the following pairs of graphs are isomorphic. If the graphs are not isomorphic, give an invariant that the graphs do not share.*

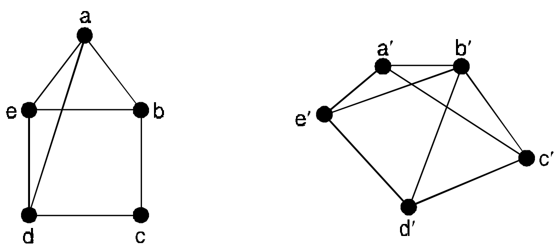
(*i*)



(*ii*)



(*iii*)



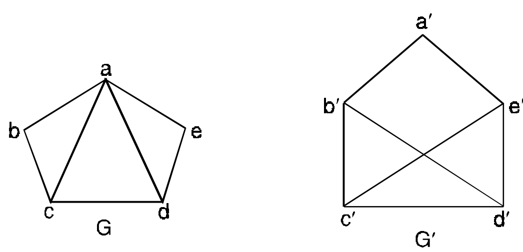
Solution. (*i*) Non isomorphic, they do not have the same number of vertices.

(*ii*) Non isomorphic, vertices of degree 3 are adjacent in one graph, non adjacent in the other.

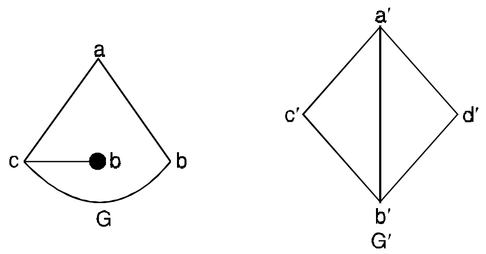
(*iii*) Non isomorphic, one has a vertex of degree 2 but other does not.

Problem 1.59. *Find whether the following pairs of graphs are isomorphic or not.*

(*i*)



(*ii*)



Solution. (*i*) Not isomorphic.

G has 2 nodes *b* and *e* of degree 2 while G′ has one node *a*′ of degree 2.

(*ii*) Not isomorphic.

G has 4 edges, and G′ has edges.

Problem 1.60. *If a graph G of n vertices is isomorphic to its complement* $\overline{G}$*, show that n or (n – 1) must be a multiple of 4.*

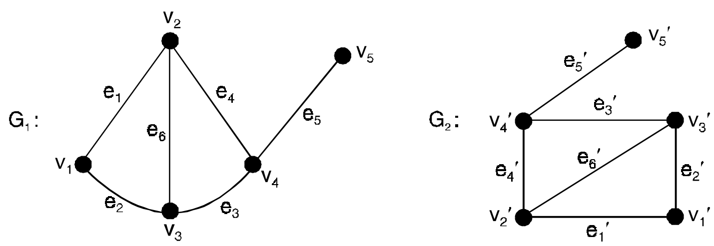
Solution. Since $G \approx \overline{G}$, both of G and $\overline{G}$ have the same number of edges.

Also, the total number of edges in G and $\overline{G}$ taken together must be equal to the number of edges in K_n.

Since K_n has $\frac{n(n-1)}{2}$ edges, it follows that each of G and $\overline{G}$ has $\frac{n(n-1)}{4}$ edges.

Thus, $\frac{n(n-1)}{4}$ must be a positive integer, as such, *n* or (*n* – 1) must be a multiple of 4.

Problem 1.61. *Consider two graphs* G_1 *and* G_2 *as shown below, show that the graphs* G_1 *and* G_2 *are isomorphic.*



Solution. The correspondence between the graphs is as follows :

The vertices (v_1, v_2, v_3, v_4, v_5) in G_1 correspond to (v_1', v_2', v_3', v_4', v_5') respectively in G_2.

The edges (e_1, e_2, e_3, e_4, e_5, e_6) in G_1 correspond to (e_1', e_2', e_3', e_4', e_5', e_6') respectively in G_2.

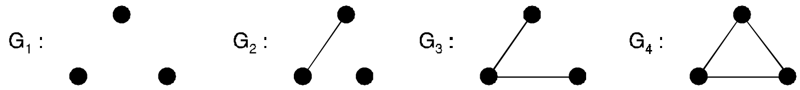
Here the incidence property is preserved.

Therefore the graphs G_1 and G_2 are isomorphic to each other.

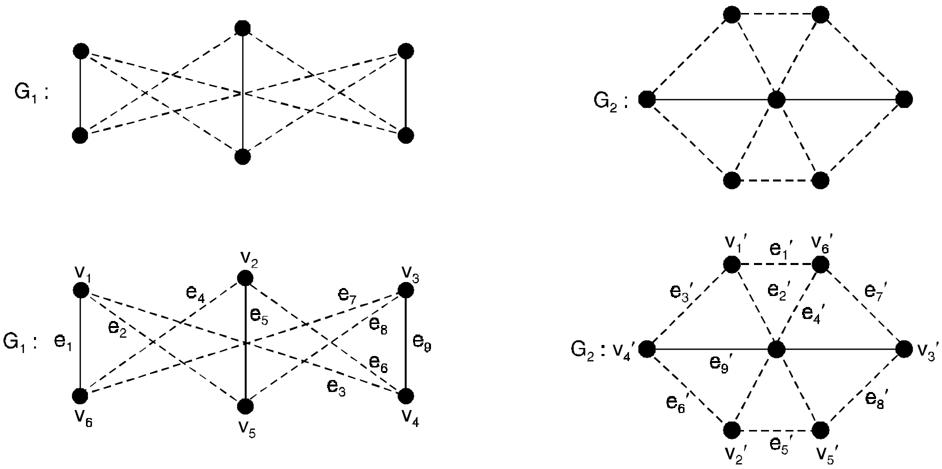
Problem 1.62. *Draw all non-isomorphic graphs on 2 and 3 vertices.*

Solution. All non-isomorphic graphs on 2 vertices are

All non-isomorphic graphs on 3 vertices are



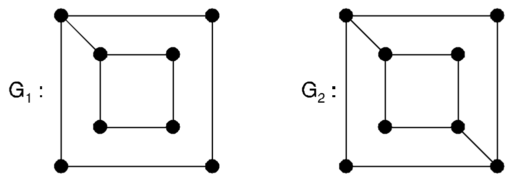
Problem 1.63. *Show that the following graphs are isomorphic.*



Solution. There is one-to-one correspondence between vertices and one-to-one correspodence between edges. Further incidence property is preserved.

Therefore G_1 is isomorphic to G_2,

Problem 1.64. *Determine whether the following graphs are isomorphic or not*



Solution. Here both the graphs G_1 and G_2 contains 8 vertices and 10 edges.

The number of vertices of degree 2 in both the graphs are four.

Also the number of vertices of degree 3 in both the graphs are four.

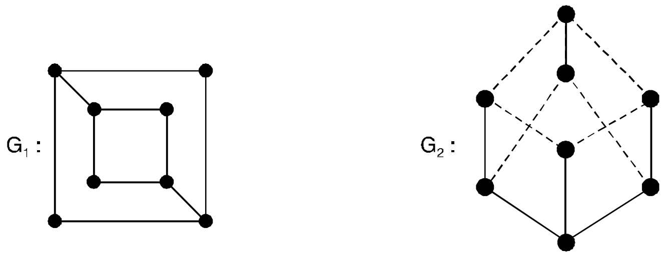
For adjacency, consider the vertex of degree 3 in G_1. It is adjacent to two vertices of degree 3 and one vertex of degree 2.

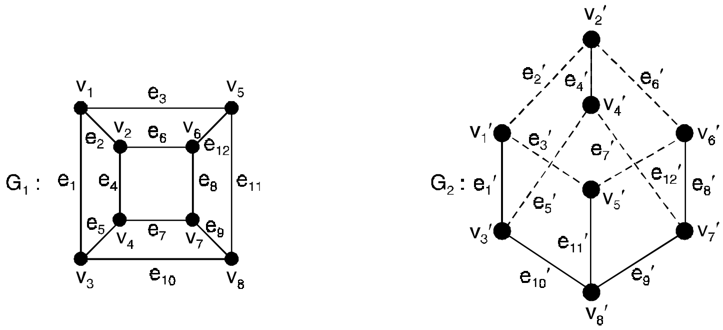
But in G_2 there does not exist any vertex of degree 3, which is adjacent to two vertices of degree 3 and one vertex of degree 2.

i.e., adjacency is not preserved.

Hence, given graphs are not isomorphic.

Problem 1.65. *Show that the following graphs are isomorphic.*

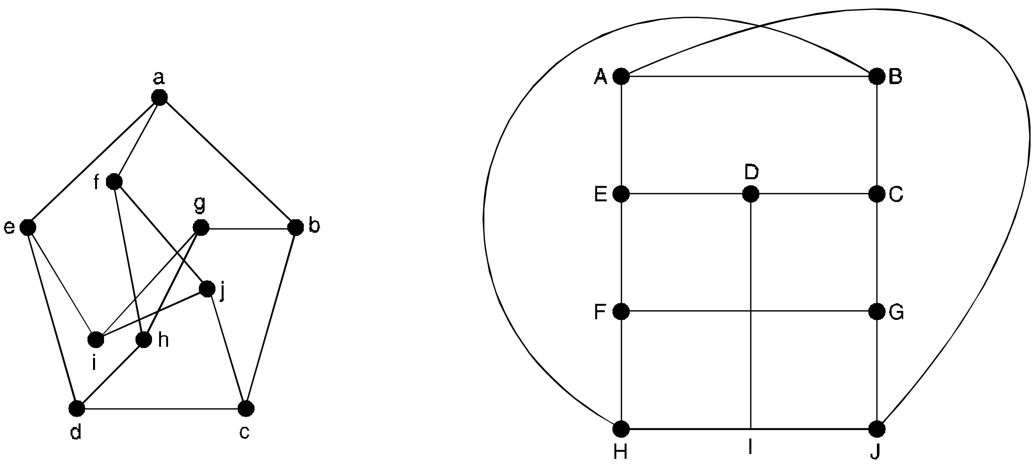




Solution. There are one-to-one correspondence between the vertices as well as between edges. Further, the incidence property is preserved.

Therefore, G_1 is isomorphic to G_2.

Problem 1.66. *Establish a one-one correspondence between the vertices and edges to show that the following graphs are isomorphic.*



Graph G_1 Graph G_2

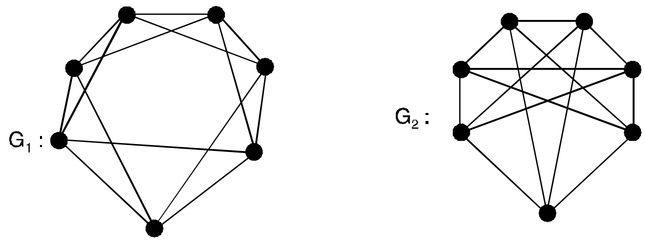
Solution. Define $\phi : V(G_1) \to V(G_2)$ by $\phi(a) = A$, $\phi(b) = B$

$\phi(c) = C$, $\phi(d) = D$, $\phi(e) = E$

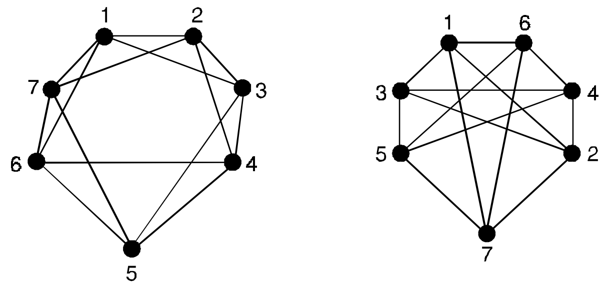
$\phi(f) = J$, $\phi(g) = H$, $\phi(h) = I$

$\phi(i) = F$, $\phi(j) = G$.

Problem 1.67. *Show that the following graphs are isomorphic.*



Solution. We first label the vertices of the graph as follows :



Define an isomorphism $\phi : V(G_1) \to V(G_2)$ by $\phi(i) = i$, we observe that ϕ preserves the adjacency and non-adjacency of the vertices.

Hence G_1 and G_2 are isomorphic to each other.

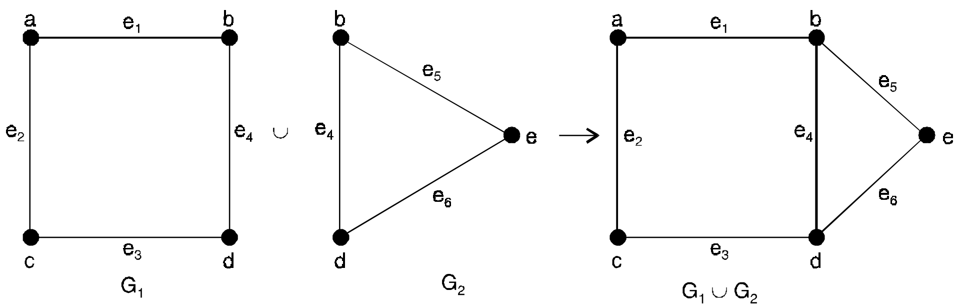
1.10 OPERATIONS OF GRAPHS

1.10.1. Union

Given two graphs G_1 and G_2, their union will be a graph such that

$$V(G_1 \cup G_2) = V(G_1) \cup V(G_2)$$

and
$$E(G_1 \cup G_2) = E(G_1) \cup E(G_2)$$

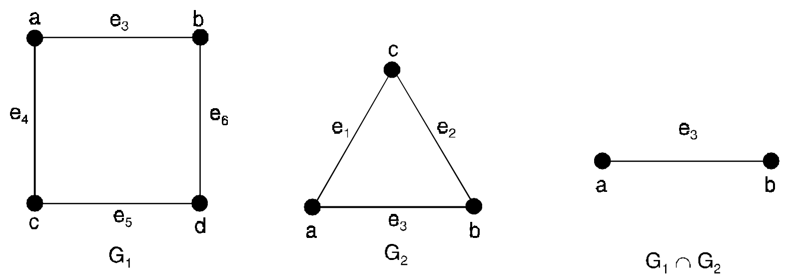


1.10.2. Intersection

Given two graphs G_1 and G_2 with at least one vertex in common then their intersection will be a graph such that

$$V(G_1 \cap G_2) = V(G_1) \cap V(G_2)$$

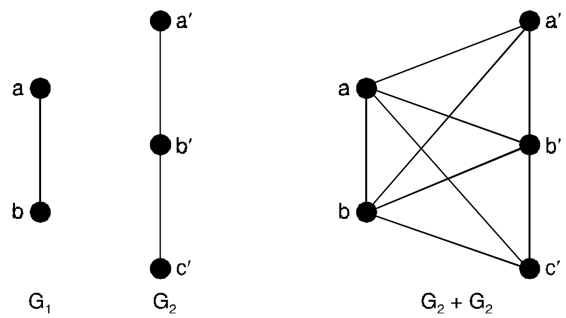
and $$E(G_1 \cap G_2) = E(G_1) \cap E(G_2)$$



1.10.3. Sum of two graphs

If the graphs G_1 and G_2 such that $V(G_1) \cap V(G_2) = \phi$, then the sum $G_1 + G_2$ is defined as the graph whose vertex set is $V(G_1) + V(G_2)$ and the edge set is consisting those edges, which are in G_1 and in G_2 and the edges obtained, by joining each vertex of G_1 to each vertex of G_2.

For example,



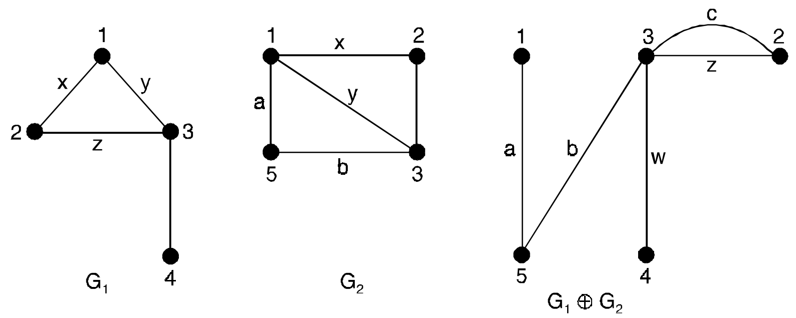
1.10.4. Ring sum

Let G_1 (V_1, E_1) and G_2 (V_2, E_2) be two graphs. Then the ring sum of G_1 and G_2, denoted by $G_1 \oplus G_2$ is defined as the graph G such that :

(*i*) $V(G) = V(G_1) \cup V(G_2)$

(*ii*) $E(G) = E(G_1) \cup E(G_2) - E(G_1) \cap E(G_2)$

i.e., the edges that either in G_1 or G_2 but not in both. The ring sum of two graphs G_1 and G_2 is shown below.



1.10.5. Product of graphs

To define the product $G_1 \times G_2$ of two graphs consider any two points $u = (u_1, u_2)$ and $v = (v_1, v_2)$ in $V = V_1 \times V_2$. Then u and v are adjacent in $G_1 \times G_2$ whenever [$u_1 = v_1$ and u_2 adj. v_2] or [$u_2 = v_1$ and u_1 adj. v_1]

For example,

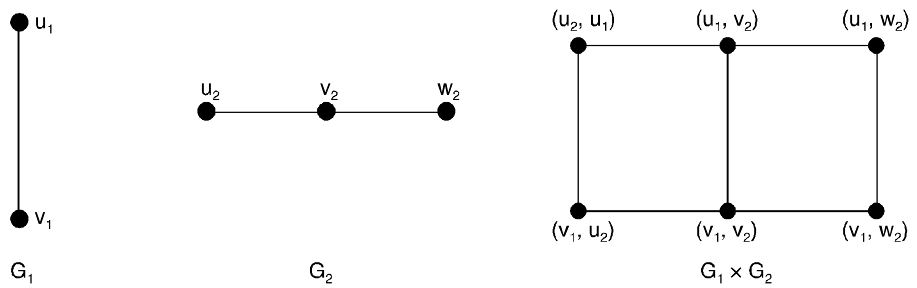


Fig. (*a*). The product of two graphs.

1.10.6. Composition

The composition $G = G_1[G_2]$ also has $V = V_1 \times V_2$ as its point set, and $u = (u_1, u_2)$ is adjacent with $v = (v_1, v_2)$ whenever (u_1 adj. v_1) or ($u_1 = v_1$ and u_2 adj. v_1)

For the graphs G_1 and G_2 of Figure. (*a*), both compositions $G_1[G_2]$ and $G_2[G_1]$ are shown in Figure (*b*).

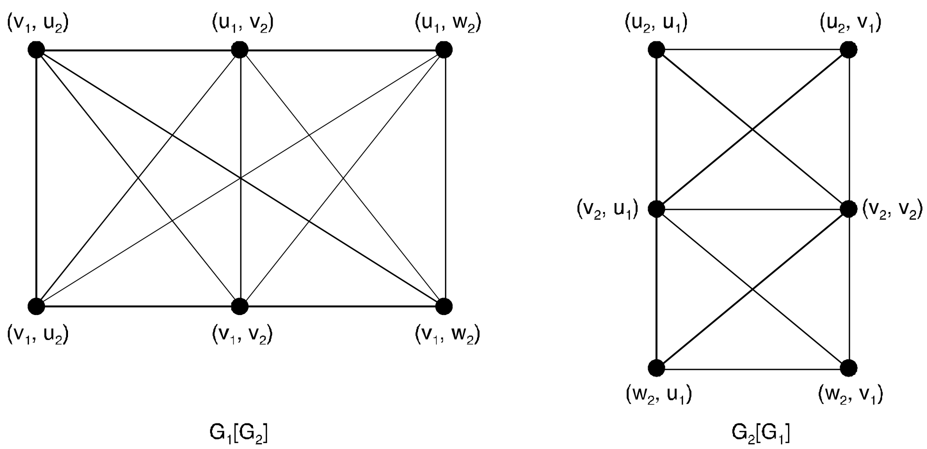
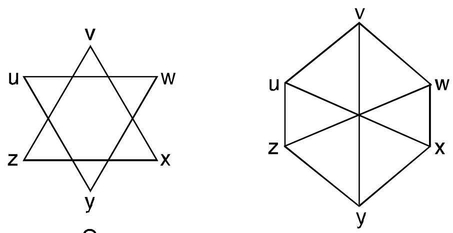


Fig. (*b*). Two compositions of graphs

1.10.7. Complement

The complement G′ of G is defined as a simple graph with the same vertex set as G and where two vertices *u* and *v* adjacent only when they are not adjacent in G.

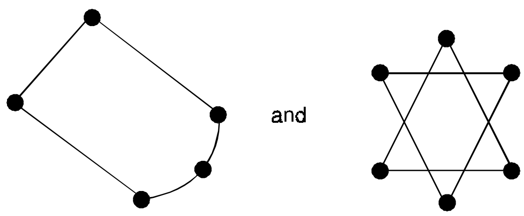
For example,



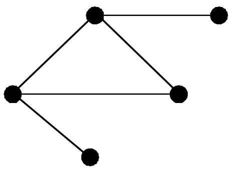
G   Complement of G.

A graph G is self-complementary if it is isomorphic to its complement.

For example, the graphs



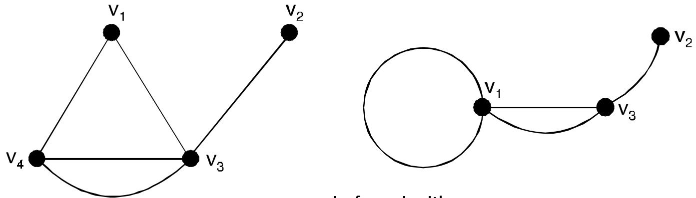
Self-complementary. The other self-complementary graph with five vertices is



1.10.8. Fusion

A pair of vertices v_1 and v_2 in graph G is said to be 'fused' if these two vertices are replaced by a single new vertex *v* such that every edge that was adjacent to either v_1 or v_2 or both is adjacent *v*.

Thus we observe that the fusion of two vertices does not alter the number of edges of graph but reduced the vertices by one.



v_1 is fused with v_4

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 $\overline{G}$.

1.11 THE PROBLEM OF RAMSEY : 1.4

Prove that at any party with six people, there are three mutual acquaintances or three mutual nonacquaintances.

Solution. This situation may be represented by a graph G with six points standing for people, in which adjacency indicates acquaintance.

Then the problem is to demonstrate that G has three mutually adjacent points or three mutually nonadjacent ones.

The complement $\overline{G}$ of a graph G also has V(G) as its point set, but two points are adjacent in $\overline{G}$ if and only if they are not adjacent in G.

In Figure 28, G has no triangles, while $\overline{G}$ consists of exactly two triangles.

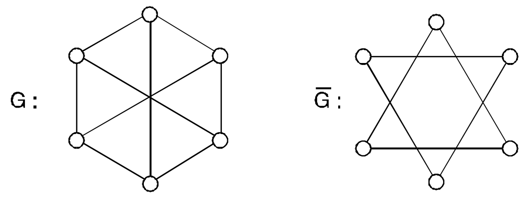


Fig. 28. A graph and its complement

In figure 29 : A self-complementary graph is isomorphic with its complement.

The complete graph K_p has every pair of its P points adjacent. Since V is not empty, P ≥ 1.

Thus K_P has $\binom{P}{2}$ lines and is regular of degree P – 1.

As we have seen, K_3 is called a triangle. The graphs $\overline{K}_P$ are totally disconnected, and are regular of degree 0.

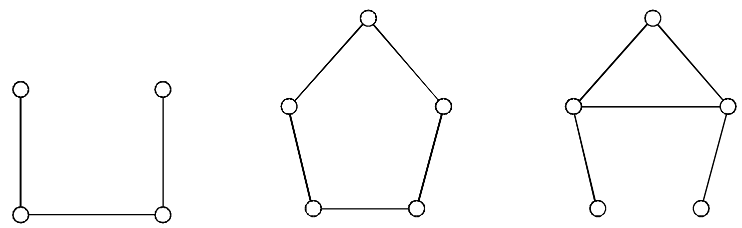


Figure 29. The smallest nontrivial self-complementary graphs.

Theorem 1.5. *The maximum number of lines among all P point graphs with no triangles is* $\left[\dfrac{P^2}{4}\right]$.

Proof. The statement is obvious for small values of P. An inductive proof may be given separately for odd P and for even P.

Suppose the statement is true for all even $P \le 2n$.

We then prove it for $P = 2n + 2$

Thus, let G be a graph with $P = 2n + 2$ points and no triangles.

Since G is not totally disconnected, there are adjacent points u and v.

The subgraph $G' = G - \{u, v\}$ has $2n$ points and no triangles, so that by the inductive hypothesis G' has at most $\left[\dfrac{4n^2}{4}\right] = n^2$ lines.

There can be no point W such that u and v are both adjacent to W, for then u, v and w would be points of a triangle in G.

Thus if u is adjacent to K points of G', v can be adjacent to at most $2n - K$ points.

Then G has at most

$$n^2 + K + (2n - K) + 1 = n^2 + 2n + 1 = \frac{P^2}{4} = \left[\frac{P^2}{4}\right] \text{ lines.}$$

Theorem 1.6. *Every graph is an intersection graph.*

Proof. For each point v_i of G

Let S_i be the union of $\{v_i\}$ with the set of lines incident with v_i.

Then it is immediate that G is isomorphic with Ω (F) where $F = \{S_i\}$.

Note : The intersection number ω' (G) of a given graph G is the minimum number of elements in a set S such that G is an intersection on S.

Corollary (1)

If G is connected and $P \ge 3$, then $\omega(G) \le q$.

Proof. In this case, the points can be omitted from the sets S_i used in the proof of the theorem, so that $S = X(G)$.

Corollary (2)

If G has P_0 isolated points and no K_2 components, then $\omega(G) \le q + P_0$.

Theorem 1.7. *Let G be a connected graph with P > 3 points. Then $\omega(G) = q$ if and only if G has no triangles.*

Proof. We first prove the sufficiency.

To show that $\omega(G) \ge q$ for any connected G with atleast 4 points having no triangles.

By definition of the intersection number, G is isomorphic with an intersection graph $\Omega(F)$ on a set S with $| S | = \omega(G)$.

For each point v_i of G, let S_i be the corresponding set.

Because G has no triangles, no element of S can belong to more than two of the sets S_i, and $S_i \cap S_j \ne \phi$ if and only if v_iv_j as a line of G.

Thus we can form a 1 – 1 correspondence between the lines of G and those elements of S which belong to exactly two sets S_i.

Therefore $\omega(G) = | S | \ge q$ so that $\omega(G) = q$.

To prove necessity :

Let $\omega(G) = q$ and assume that G has a triangle then let G_1 be a maximal triangle-free spanning subgraph of G. $\omega(G_1) = q_1 = | X(G_1) |$.

Suppose that $G_1 = \Omega(F)$, where F is a family of subsets of some set S with cardinality q_1.

Let x be a line of G not in G_1 and consider $G_2 = G_1 + x$. Since G_1 is a maximal triangle-free, G_2 must have some triangle, say u_1, u_2, u_3 where $x = u_1u_3$.

Denote by S_1, S_2, S_3 the subsets of S corresponding to u_1, u_2, u_3. Now if u_2 is adjacent to only u_1 and u_3 in G_1, replace S_2 by a singleton chosen from $S_1 \cap S_2$ and add that element to S_3.

Otherwise, replace S_3 by the union of S_3 and any element in $S_1 \cap S_2$.

In either case this gives a famly F′ of distinct subsets of S such that $G_2 = \Omega(F')$.

Thus $\omega(G_2) \le q_1$ while $| X(G_2) | = q_1 + 1$

If $G_2 \cong G$ there is nothing to prove.

But if $G_2 \ne G$, then let $| X(G) | - | X(G_2) | = q_0$

It follows that G is an intersection graph on a set with $q_1 + q_0$ elements.

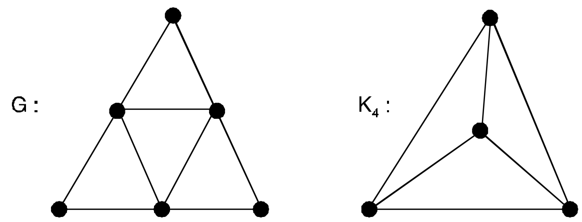
However, $q_1 + q_0 = q - 1$

Thus $\omega(G) < q$

Hence the proof.

Theorem 1.8. *For any graph G with $P \ge 4$ points, $\omega(G) \le \left[\dfrac{P^2}{4}\right]$.*

Theorem 1.9. *A graph G is a clique graph if and only if it contains a family F of complete subgraphs, whose union in G, such that whenever every pair of such complete graphs in some subfamily F ′ have a non empty intersection, the intersection of all the members of F ′ is non empty.*



A graph and its clique graph.

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.

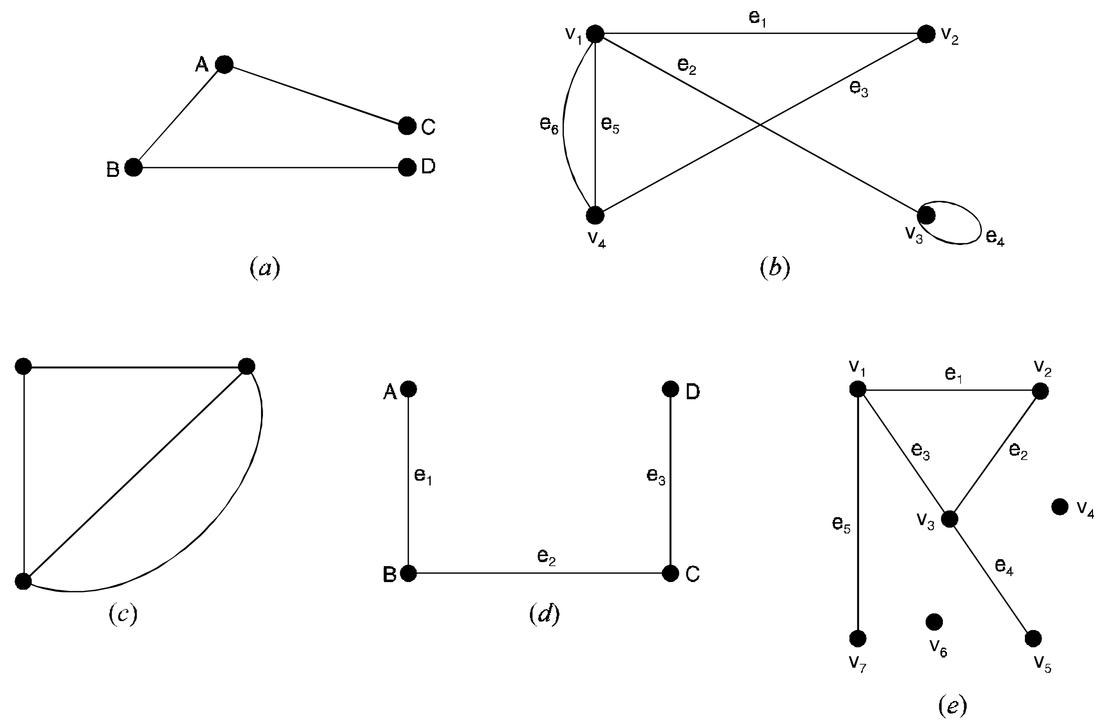


Fig. 30.

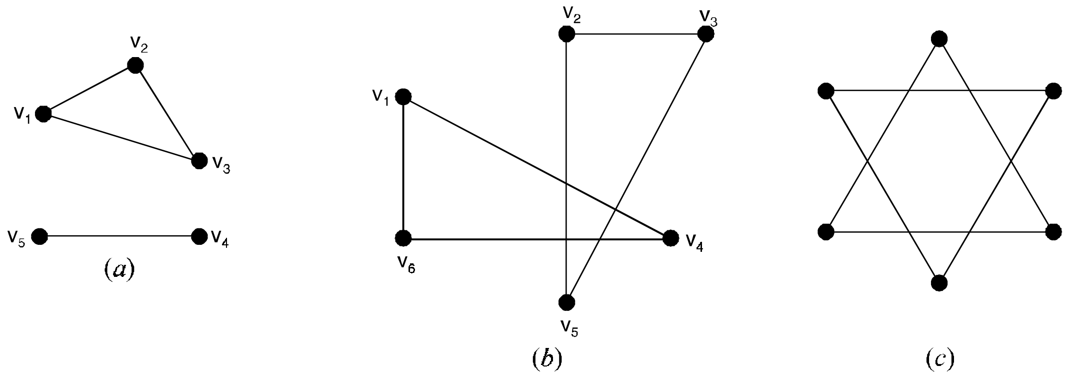


Fig. 31.

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.

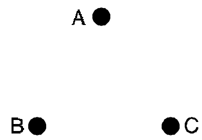


Fig. 32.

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.

For example, the graphs in Figure 31(*a, b*) have two components each.

1.12.1. Path graphs and cycle graphs

A connected graph that is 2-regular is called a cycle graph. Denote the cycle graph of *n* vertices by Γ_n. A circuit in a graph, if it exists, is a cycle subgraph of the graph.

The graph obtained from Γ_n by removing an edge is called the path graph of *n* vertices, it is denoted by P_n.

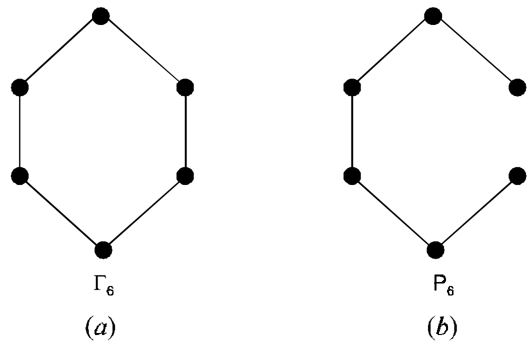


Fig. 33.

The graphs Γ_6 and P_6 are shown in Figure 33(*a*) and 33(*b*) respectively.

1.12.2. Rank and nullity

For a graph G with n vertices, m edges and k components we define the rank of G and is denoted by $\rho(G)$ and the nullity of G is denoted by $\mu(G)$ as follows.

$$\rho(G) = \text{Rank of G} = n - k$$

$$\mu(G) = \text{Nullity of G} = m - \rho(G) = m - n + k$$

If G is connected, then we have

$$\rho(G) = n - 1 \text{ and } \mu(G) = m - n + 1.$$

Problem 1.68. *Prove that a simple graph with n vertices must be connected if it has more than* $\frac{(n-1)(n-2)}{2}$ *edges.*

Solution. Consider a simple graph on n vertices.

Choose $n - 1$ vertices $v_1, v_2, ..., v_{n-1}$ of G.

We have maximum ${}^{n-1}C_2 = \frac{(n-1)(n-2)}{2}$ number of edges only can be drawn between these vertices.

Thus if we have more than $\frac{(n-1)(n-2)}{2}$ edges atleast one edge should be drawn between the nth vertex v_n to some vertex v_i, $1 \le i \le n - 1$ of G.

Hence G must be connected.

Problem 1.69. *Show that if a and b are the only two odd degree vertices of a graph G, then a and b are connected in G.*

Solution. If G is connected, nothing to prove.

Let G be disconnected.

If possible assume that a and b are not connected.

Then a and b lie in the different components of G.

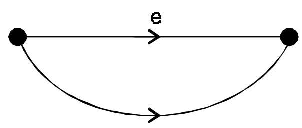
Hence the component of G containing a (similarly containing b) contains only one odd degree vertex a, which is not possible as each component of G is itself a connected graph and in a graph number of odd degree vertices should be even.

Therefore a and b lie in the same component of G.

Hence they are connected.

Problem 1.70. *Prove that a connected graph G remains connected after removing an edge e from G if and only if e lie in some circuit in G.*

Solution. If an edge e lies in a circuit C of the graph G then between the end vertices of e, there exist atleast two paths in G.



Hence removal of such an edge *e* from the connected graph G will not effect the connectivity of G.

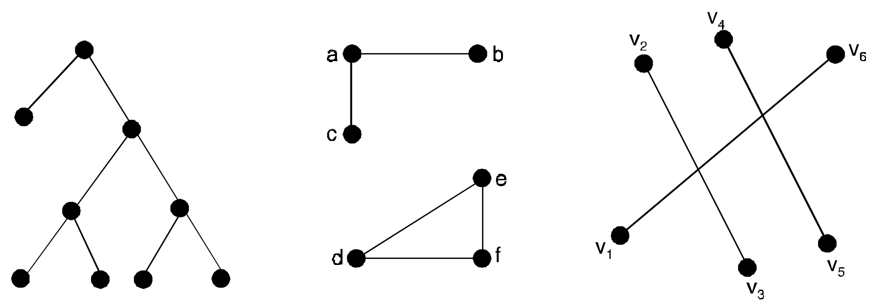
Conversely, if *e* does not lies in any circuit of G then removal of *e* disconnects the end vertices of *e*. Hence G is disconnected.

Problem 1.71. *If G_1 and G_2 are (edge) decomposition of a connected graph G, then prove that $V(G_1) \cap V(G_2) \neq \phi$.*

Solution. If $V(G_1) \cap V(G_2) = \phi$ then $V(G_1)$ and $V(G_2)$ are the vertex partition of V(G) (there exists no edges left in G to include between vertex of $V(G_1)$ and $V(G_2)$ as G_1 and G_2 are edge partition of G).

Hence, G is disconnected, a contradiction to the fact that G is connected.

Problem 1.72. *Which of the graphs below are connected :*

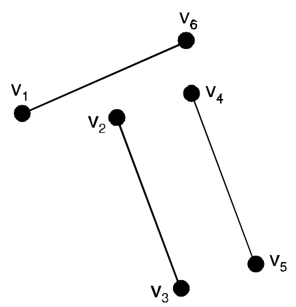


(*a*) (*b*) (*c*)

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

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

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



Theorem 1.10. *If a graph (connected or disconnected) has exactly two vertices of odd degree, there must be a path joining these two vertices.*

Proof. Let G be a graph with all even vertices except vertices v_1 and v_2, which are odd.

From theorem, which holds for every graph and therefore for every component of a diconnected graph, No graph can have an odd number of odd vertices.

Therefore, in graph G, v_1 and v_2 must belong to the same component and hence must have a path between them.

Theorem 1.11. *A simple graph with n vertices and k components cannot have more than* $\frac{(n-k)(n-k+1)}{2}$ *edges.*

Proof. Let n_i = the number of vertices in component i,

$$1 \le i \le k, \qquad \text{then } \sum_{i=1}^{k} n_i = n.$$

A component with n_i vertices will have the maximum possible number of edges when it is complete.

That is, it will contain $\frac{1}{2} n_i(n_i - 1)$ edges.

Hence the maximum number of edges is :

$$\frac{1}{2}\sum_{i=1}^{k} n_i\,(n_i - 1) = \frac{1}{2}\sum_{i=1}^{k} n_i^2 - \frac{1}{2}\sum_{i=1}^{k} n_i$$

$$\le \frac{1}{2}\,[n^2 - (k-1)(2n-k)] - \frac{1}{2}\,n$$

$$= \frac{1}{2}\,[n^2 - 2nk + k^2 + n - k]$$

$$= \frac{1}{2}\,(n-k)(n-k+1).$$

Corollary :

If $m > \frac{1}{2}\,(n-1)(n-2)$ then a simple graph with n vertices and m edges is connected.

Proof. Suppose the graph is disconnected. Then it has at least two components, therefore by theorem.

$$m \le \frac{1}{2}\,(n-k)(n-k+1) \text{ for } k \ge 2$$

$$\le \frac{1}{2}\,(n-2)(n-1)$$

This contradicts the assumption that $m > \frac{1}{2}\,(n-1)(n-2)$.

Therefore, the graph should be connected.

Theorem 1.12. *A graph G is disconnected if and only if its vertex set V can be partitioned into two subsets V_1 and V_2 such that there exists no edge in G whose one end vertex is in the subset V_1 and the other in the subset V_2.*

Proof. Let G be disconnected. Then we have by the definition that there exists a vertex x in G and a vertex y in G such that there is no path between x and y in G

Let $V_1 = \{Z \in V : z$ is connected to $x\}$. Then V_1 is the set of all vertices of G which are connected to x.

Let $V_2 = V - V_1$. Then $V_1 \cap V_2 = \phi$ and $V_1 \cup V_2 = V$.

Hence V_1 and V_2 are the partition of V(G). Let a be any vertex of V_1.

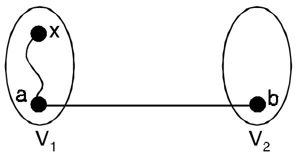
To prove that 'a' is not adjacent to any vertex of V_2.

If possible let $b \in V_2$ such that $ab \in E(G)$. Then $a \in V_1$ there exist a path P_1 : from x to a.

This path can be extended to the path $P_2 = P_1, ab, b$.

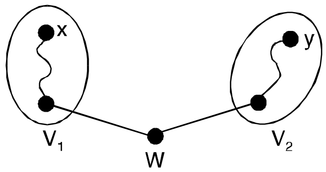
P_2 is a path from x to b in G.

Therefore x and b are connected. This implies that $b \in V_1$ which is contradiction to the fact $V_1 \cap V_2 = \phi$.



Conversely, let us assume that V can be partitioned into two subsets V_1 and V_2 such that no vertex of V_1 is adjacent to a vertex of V_2.

Let x be any vertex in V_1 and y be any vertex in V_2.



To prove that G is disconnected, if possible, suppose G is connected. Then x and y are connected.

Therefore, there exists a path between x and y in G. But this path is possible only through a vertex W in G which is not either in V_1 or V_2.

Hence $V_1 \cup V_2 \neq V$, a contradiction.

Theorem 1.13. *Show that a simple (p, q)-graph is connected then $P \leq q + 1$.*

Proof. The proof is by induction on the number of edges in G. If G has only one or two edges then the theorem is true. Assume that the theorem is true for each graph with fewer than n edges.

Let G be given connected (p, q) graph.

Case (*i*) : G contains a circuit.

Let S be a graph obtained by G by removing an edge from a circuit of G. Then S is a connected graph having $q - 1$ edges. The number of vertices of S and G are same, hence by inductive hypothesis $p \leq q - 1 + 1$.

Thus $p \leq q$, hence certainly $p \leq q + 1$.

Case (*ii*) : G does not contain a circuit.

Let p be a longest path in G. Let a and b be the end vertices of the path. The vertex a must be of degree 1, otherwise the path could be made longer, or there would be a circuit in G.

Remove the vertex a and the edge incident with the vertex a.

Let H be the graph so obtained. Then H contains exactly one vertex and one edge less than that of G.

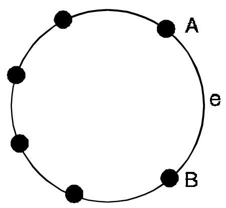
Further H is connected, hence by inductive hypothesis $p - 1 \leq (q - 1) + 1$.

Hence $p \leq q + 1$.

Problem 1.73. *Prove that a connected graph G remains connected after removing an edge e from G if and only if e belongs to some circuit in G.*

Solution. Suppose e belongs to some circuit C in G. Then the end vertices of e, say, A and B are joined by atleast two paths, one of which is e and the other C – e.

Hence the removal of e from G will not affect the connectivity of G ; even after the removal of e the end vertices of e. (*i.e.,* A and B) remain connected.



Conversely, suppose e does not belong to any circuit in G. Then the end vertices of e are connected by atmost one path.

Hence the removal of e from G disconnects these end points. This means that G – e is a disconnected graph.

Thus, if e does not belong to any circuit in G then G – e is disconnected.

This is equivalent to saying that if G – e is connected then e belongs to some circuit in G.

Problem 1.74. *Let G be a disconnected graph with n vertices where n is even. If G has two components each of which is complete, prove that G has a minimum of* $\frac{n(n-2)}{4}$ *edges.*

Solution. Let x be the number of vertices in one of the components.

Then the other component has $n - x$ number of vertices since both components are complete graphs, the number of edges they have are $\frac{x(x-1)}{2}$ and $\frac{(n-x)(n-x-1)}{2}$ respectively.

Therefore, the total number of edges in G is

$$m = \frac{x(x-1)}{2} + \frac{(n-x)(n-x-1)}{2}$$

$$= x^2 - nx + \frac{n}{2}(n-1)$$

$$\Rightarrow \quad m' = 2x - n,\ m'' = 2 > 0, \qquad \left(m' = \frac{dm}{dx} \text{ and } m'' \frac{d^2m}{dx^2}\right)$$

Therefore, m is minimum when $2x - n = 0$

$$\Rightarrow \quad x = \frac{n}{2}$$

$$\text{Min. } m = \left(\frac{n}{2}\right)^2 - n\left(\frac{n}{2}\right) + \frac{n}{2}(n-1)$$

$$= \frac{n(n-2)}{4}.$$

Problem 1.75. *Find the rank and nullity of the complete graph k_n.*

Solution. k_n is a connected graph with n vertices and

$$m = \frac{n(n-1)}{2} \text{ edges.}$$

Therefore, by the definitions of rank and nullity, we have

Rank of $k_n = n - 1$

$$\text{Nullity of } k_n = m - n + 1 = \frac{1}{2}n(n-1) - n + 1$$

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

1.13 WALKS, PATHS AND CIRCUITS

1.13.1. Walk

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

$$v_i e_j,\ v_{i+1}\ e_{j+1},\ v_{i+2},\ \ldots\ldots,\ e_k v_m$$

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

$$v_2e_4v_6e_5v_4e_3v_3 \text{ and } v_1e_8v_2e_4v_6e_6v_5e_7v_5 \text{ are walks.}$$

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.

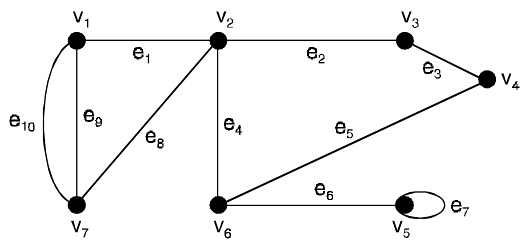


Fig. 34.

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 $\boldsymbol{u - v}$ **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.

1.13.2. Path

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.

1.13.3. Circuit

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.

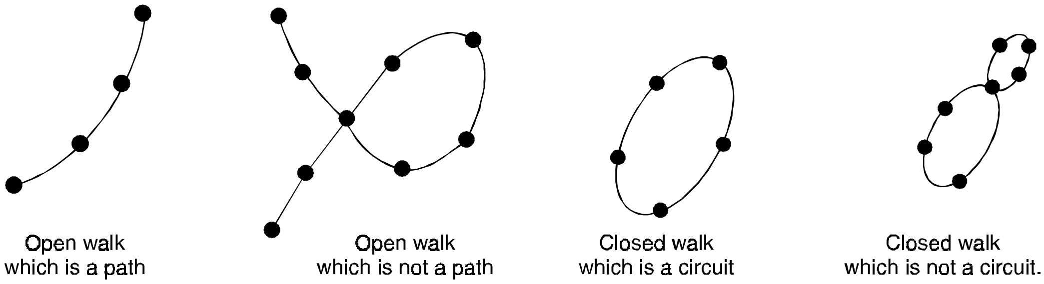
But $v_1e_9v_7e_8v_2e_4v_6e_5v_4e_3v_3e_2v_2e_1v_1$ is a closed walk but not a circuit.

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.



1.13.4. Length

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).

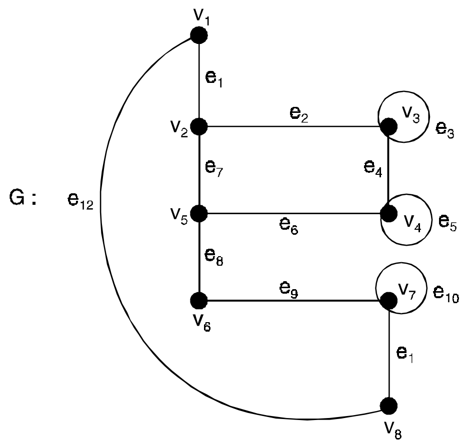


Fig. 35.

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.

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

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

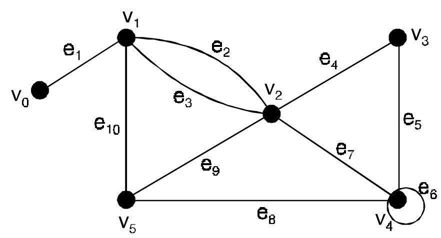
$W_3 : v_3e_4v_4e_5v_4e_6v_5$, $l(W_3) = 3$.

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 $\pi : u, v_1, v_2, v_3, \ldots.. v_k, v$ 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

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

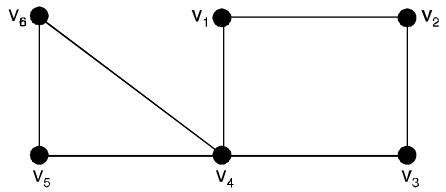
Deleting the sequence of vertices that leads back to the node v_j, all the cycles in the path can be removed.

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.

Problem 1.79. *Let G = (V, E) be a loop-free connected undirected graph, and let {a, b} be an edge of G. Prove that {a, b} is part of a cycle if and only if its removal (the vertices a and b are left) does not disconnect G.*

Solution. If $\{a, b\}$ is not part of a cycle, then its removal disconnects a and b (and G).

If not, there is a path P from a to b, and P together with $\{a, b\}$ provides a cycle containing $\{a, b\}$.

Conversely, if the removal of $\{a, b\}$ from G disconnects G, there exist $x, y \in$ V such that the only path P from x to y contains $e = \{a, b\}$. If e were part of a cycle C, then the edges in $(P - \{e\}) \cup (C - \{e\})$ would contain a second path connecting x to y.

Theorem 1.16. *In a graph G, every $u - v$ path contains a simple $u - v$ path.*

Proof. If a path is a closed path, then it certainly contains the trivial path.

Assume, then, that P is an open $u - v$ path.

We complete the proof by induction on the length n of P.

If P has length one, then P is itself a simple path.

Suppose that all open $u - v$ paths of length k. Where $1 \le k \le n$, contains a simple $u - v$ path. Now suppose that P is the open $u - v$ path

$\{v_0, v_1\}$, ... $\{v_n, v_{n+1}\}$, where $u = v_0$ and $v = v_{n+1}$ of course, it may be that P has repeated vertices, but if not, then P is a simple $u - v$ path.

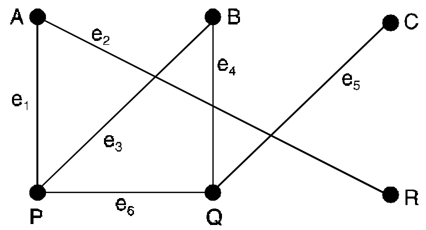
If, on the other hand, there are repeated vertices in P.

Let i and j be distinct positive integers where $i < j$ and $v_i = v_j$.

If the closed path $v_i - v_j$ is removed from P, an open path P′ is obtained having length $\le n$, since at least the edge $\{v_i, v_{i+1}\}$ was deleted from P.

Thus, by the inductive hypothesis, P′ contains a simple $u - v$ path and, thus, so does P.

Problem 1.80. *Find all circuits in the graph shown below :*



Solution. There are no circuits beginning and ending with the vertices A, C and R.

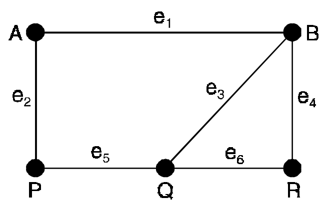
The circuits beginning and ending with the vertices

B, P, Q are Be_3Pe_6Qe_4B, Pe_6Qe_4Be_3P, Qe_4Be_3Pe_6Q

But all of these represent one and the same circuit.

Thus, there is only one circuit in the graph.

Problem 1.81. *Consider the graph shown in Figure, find all paths from vertex A to vertex R. Also, indicate their lengths.*



Solution. There are four paths from A to R.

These are Ae_1Be_4R, Ae_1Be_3Qe_6R, Ae_2Pe_5Qe_6R, Ae_2Pe_5Qe_3Be_4R.

These paths contain, two, three the and four edges.

Their lengths are two, three, three and four respectively.

Problem 1.82. *Prove the following :*

(*a*) *A path with n vertices is of length n – 1*

(*b*) *If a circuit has n vertices, it has n edges*

(*c*) *The degree of every vertex in a circuit is two.*

Solution. (*a*) In a path, every vertex except the last is followed by precisely one edge.

Therefore, if a path has n vertices, it must have $n - 1$ edges. Its length is $n - 1$.

(*b*) In a circuit, every vertex is followed by precisely one edge.

Therefore, if a circuit has n vertices, it must have n edges.

(*c*) In a circuit, exactly two edges are incident on every vertex.

Therefore, the degree of every vertex in a circuit is two.

Problem 1.83. *If G is a simple graph in which every vertex has degree at least k, prove that G contains a path of length at least k. Deduce that if $k \geq 2$ then G also contains a circuit of length at least $k + 1$.*

Solution. Consider a path P in G, which has a maximum number of vertices. Let u be an end vertex of P. Then every neighbour of u belongs to P. Since u has at least k neighbours and since G is simple, P must have at least k vertices other than u.

Thus, P is a path of length at least k

If $k \geq 2$ then P and the edge from u to its farthest neighbour v constitute a circuit of length at least $k + 1$.

1.14 EULERIAN GRAPHS

1.14.1. Euler path

A path in a graph G is called Euler path if it includes every edges exactly once. Since the path contains every edge exactly once, it is also called Euler trail.

1.14.2. Euler circuit

An Euler path that is circuit is called Euler circuit. A graph which has a Eulerian circuit is called an Eulerian graph.

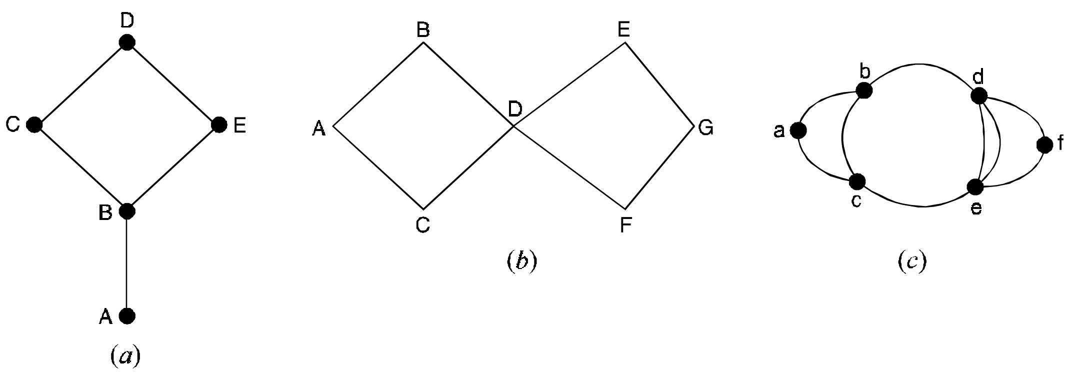


Fig. 36.

The graph of Figure 36(*a*) has an Euler path but no Euler circuit. Note that two vertices A and B are of odd degrees 1 and 3 respectively. That means AB can be used to either arrive at vertex A or leave vertex A but not for both.

Thus an Euler path can be found if we start either from vertex A or from B.

ABCDEB and BCDEBA are two Euler paths. Starting from any vertex no Euler circuit can be found.

The graph of Figure 36(*b*) has both Euler circuit and Euler path. ABDEGFDCA is an Euler path and circuit. Note that all vertices of even degree.

No Euler path and circuit is possible in Figure 36(*c*).

Note that all vertices are not even degree and more than two vertices are of odd degree.

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

Note : To determine whether a graph G has an Euler circuit, we note the following points :

(*i*) List the degree of all vertices in the graph.

(*ii*) If any value is zero, the graph is not connected and hence it cannot have Euler path or Euler circuit.

(*iii*) If all the degrees are even, then G has both Euler path and Euler circuit.

(*iv*) If exactly two vertices are odd degree, then G has Euler path but no Euler circuit.

Theorem 1.17. *The following statements are equivalent for a connected graph G :*

(*i*) *G is Eulerian*

(*ii*) *Every point of G has even degree*

(*iii*) *The set of lines of G be partitioned into cycles.*

Proof. (*i*) implies (*ii*)

Let T be an Eulerian trail in G.

Each occurrence of a given point in T contributes 2 to the degree of that point, and since each line of G appears exactly once in T, every point must have even degree.

(*ii*) implies (*iii*)

Since G is connected and non trivial, every point has degree at least 2, so G contains a cycle Z.

The removal of the lines of Z results in a spanning subgraph G_1 in which every point still has even degree.

If G_1 has no lines, then (*iii*) already holds ; otherwise, repetition of the argument applied to G_1 results in a graph G_2 in which again all points are even, etc.

When a totally disconnected graph G_n is obtained, we have a partition of the lines of G into *n* cycles.

(*iii*) implies (*i*)

Let Z_1 be one of the cycles of this partition.

If G consists only of this cycle, then G is obviously Eulerian.

Otherwise, there is another cycle Z_2 with a point *v* in common with Z_1.

The walk beginning at *v* and consisting of the cycles Z_1 and Z_2 in succession is a closed trail containing the lines of these two cycles.

By continuing this process, we can construct a closed trail containing all lines of G.

Hence G is Eulerian.

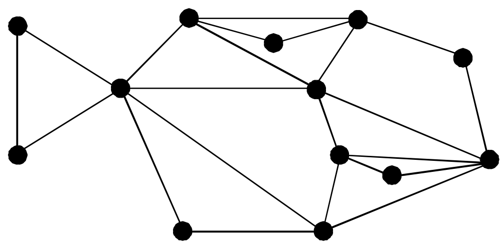


Fig. 37. An Eulerian graph.

For example, the connected graph of Figure 37 in which every point has even degree has an Eulerian trail, and the set of lines can be partitioned into cycles.

Corollary (1) :

Let G be a connected graph with exactly $2n$ odd points, $n \geq 1$, then the set of lines of G can be partitioned into n open trails.

Corollary (2) :

Let G be a connected graph with exactly two odd points. Then G has an open trail containing all the points and lines of G (which begins at one of the odd points and ends at the other).

Problem 1.18. *A non empty connected graph G is Eulerian if and only if its vertices are all of even degree.*

Proof. Let G be Eulerian.

Then G has an Eulerian trail which begins and ends at u, say.

If we travel along the trail then each time we visit a vertex we use two edges, one in and one out.

This is also true for the start vertex because we also ends there.

Since an Eulerian trial uses every edge once, each occurrence of v represents a contribution of 2 to its degree.

Thus deg(v) is even.

Conversely, suppose that G is connected and every vertex is even.

We construct an Eulerian trail. We begin a trail T_1 at any edge e. We extend T_1 by adding an edge after the other.

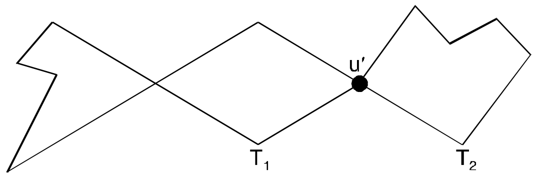
If T_1 is not closed at any step, say T_1 begins at u but ends at $v \neq u$, then only an odd number of the edges incident on v appear in T_1.

Hence we can extend T_1 by another edge incident on v.

Thus we can continue to extend T_1 until T_1 returns to its initial vertex u.

i.e., until T_1 is closed.

If T_1 includes all the edges of G then T_1 is an Eulerian trail.



Suppose T_1 does not include all edges of G.

Consider the graph H obtained by deleting all edges of T_1 from G.

H may not be connected, but each vertex of H has even degree since T_1 contains an even number of the edges incident on any vertex.

Since G is connected, there is an edge e' of H which has an end point u' in T_1.

We construct a trail T_2 in H beginning at u' and using e'. Since all vertices in H have even degree.

We can continue to extent T_2 until T_2 returns to u' as shown in Figure.

We can clearly put T_1 and T_2 together to form a larger closed trail in G.

We continue this process until all the edges of G are used.

We finally obtain an Eulerian trail, and so G is Eulerian.

Theorem 1.18. *A connected graph G has an Eulerian trail if and only if it has at most two odd vertices.*

i.e., it has either no vertices of odd degree or exactly two vertices of odd degree.

Proof. Suppose G has an Eulerian trail which is not closed. Since each vertex in the middle of the trail is associated with two edges and since there is only one edge associated with each end vertex of the trail, these end vertices must be odd and the other vertices must be even.

Conversely, suppose that G is connected with atmost two odd vertices.

If G has no odd vertices then G is Euler and so has Eulerian trail.

The leaves us to treat the case when G has two odd vertices (G cannot have just one odd vertex since in any graph there is an even number of vertices with odd degree).

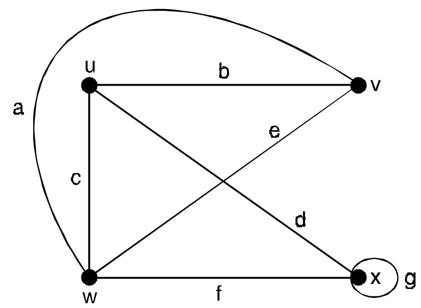
Corollary (1) :

A directed multigraph G has an Euler path if and only if it is unilaterally connected and the in degree of each vertex is equal to its out degree with the possible exception of two vertices, for which it may be that the in degree of is larger than its out degree and the in degree of the other is oneless than its out degree.

Corollary (2) :

A directed multigraph G has an Euler circuit if and only if G is unilaterally connected and the indegree of every vertex in G is equal to its out degree.

Problem 1.84. *Show that the graph shown in Figure has no Eulerian circuit but has a Eulerian trail.*

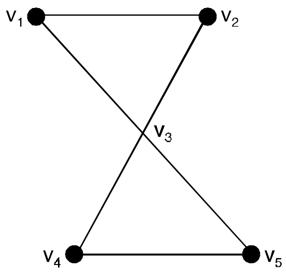


Solution. Here deg (u) = deg (v) = 3 and deg (w) = 4, deg (x) = 4

Since u and v have only two vertices of odd degree, the graph shown in Figure, does not contain Eulerian circuit, but the path.

$b - a - c - d - g - f - e$ is an Eulerian path.

Problem 1.85. *Let G be a graph of Figure. Verify that G has an Eulerian circuit.*



Solution. We observe that G is connected and all the vertices are having even degree

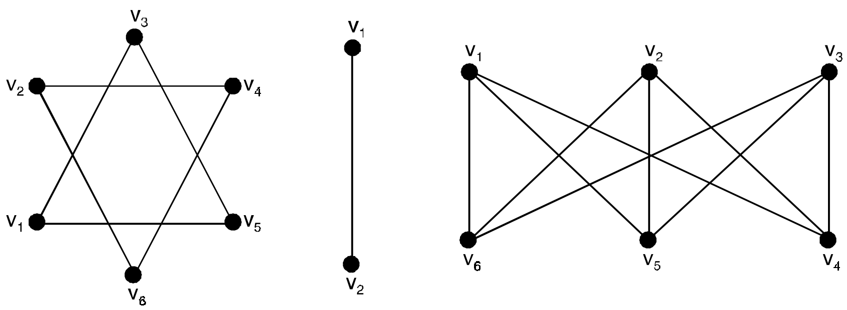
$$\deg(v_1) = \deg(v_2) = \deg(v_4) = \deg(v_5) = 2.$$

Thus G has a Eulerian circuit.

By inspection, we find the Eulerian circuit

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

Problem 1.86. *Show that the graphs in Figure below contain no Eulerian circuit.*



(a) (b) (c)

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

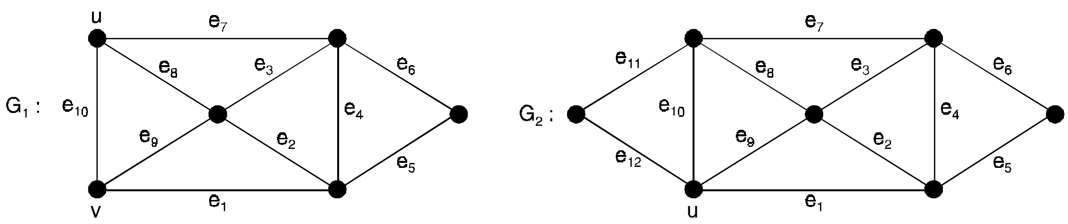
The graph shown in Figure (b) is connected but vertices v_1 and v_2 are of degree 1.

Hence it does not contain Eulerian circuit.

All the vertices of the graph shown in Figure (c) are of degree 3.

Hence it does not contain Eulerian circuit.

Problem 1.87. *Which of the following graphs have Eulerian trail and Eulerian circuit.*



Solution. In G_1 an Eulerian trail from u to v is given by the sequence of edges e_1, e_2, ..., e_{10}.

While in G_2 an Eulerian cycle (circuit) from u to v is given by e_1, e_2, ... e_{11}, e_{12}.

Problem 1.88. *Show that a connected graph with exactly two odd vertices is a unicursal graph.*

Solution. Suppose A and B be the only two odd vertices in a connected graph G.

Join these vertices by an edge *e*.

Then A and B become even vertices.

Since all other vertices in G are of even degree, the graph $G \cup e$ is an Eulerian graph.

Therefore, it has an Euler line which must include. The open walk got by deleting *e* from this Euler line is a semi-Euler line in G.

Hence G is a unicursal graph.

Problem 1.89. (*i*) *Is there is an Euler graph with even number of vertices and odd number of edges ?*

(*ii*) *Is there an Euler graph with odd number of vertices and even number of edges ?*

Solution. (*i*) Yes. Suppose C is a circuit with even number of vertices.

Let *v* be one of these vertices.

Consider a circuit C′ with odd number of vertices passing through *v* such that C and C′ have no edge in common.

The closed walk *q* that consists of the edges of C and C′ is an Eulerian graph of the desired type.

(*ii*) Yes, in (*i*), suppose C and C′ are circuits with odd number of vertices.

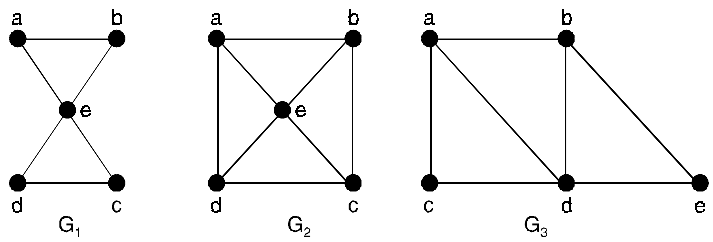
Then *q* is an Eulerian graph of the desired type.

Problem 1.90. *Find all positive integers n such that the complete graph k_n is Eulerian.*

Solution. In the complete graph k_n, the degree of every vertex is $n - 1$.

Therefore, k_n is Eulerian if and only if $n - 1$ is even, *i.e.*, if and only if *n* is odd.

Problem 1.91. *Which of the undirected graph in Figure have an Euler circuit ? Of those that do not, which have an Euler path ?*



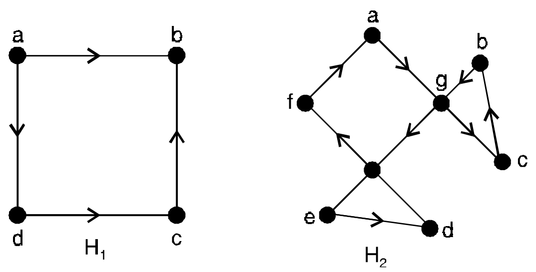
The undirected graphs G_1, G_2 and G_3.

Solution. The graph G_1 has an Euler circuit.

For example, *a, e, c, d, e, b, a.* Neither of the graphs G_2 or G_3 has an Euler circuit. However, G_3 has an Euler path, namely *a, c, d, e, b, d, a, b.*

G_2 does not have an Euler path.

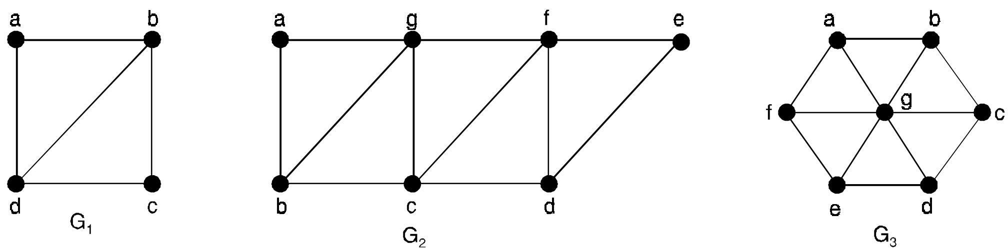
Problem 1.92. *Which of the directed graphs in Figure have an Euler circuit ? Of those that do not, which have an Euler path ?*



The directed graphs H_1, H_2, H_3

Solution. The graph H_2 has an Euler circuit, for example *a, g, c, b, g, e, d, f, a.* Neither H_1 nor H_3 has an Euler circuit. H_3 has an Euler path, namely *e, a, b, c, d, b* but H_1 does not.

Problem 1.93. *Which graphs shown in Figure have an Euler path ?*



Three undirected graphs.

Solution. G_1 contains exactly two vertices of odd degree, namely, *b* and *d*.

Hence it has an Euler path that must have *b* and *d* as its end points. One such Euler path is *d, a, b, c, d, b.* Similarly, G_2 has exactly two vertices of odd degree, namely, *b* and *d*. So it has an Euler path that must have *b* and *d* as enpoints. One such Euler path is *b, a, g, f, e, d, c, g, b, c, f, d.*

G_3 has no Euler path since it has six vertices of odd.

Lemma

If G is a graph in which the degree of each vertex is at least 2, then G contains a cycle.

Proof. If G has any loops or multiple edges, the result is trivial.

Suppose that G is a simple graph.

Let *v* be any vertex of G.

We construct a walk $v \to v_1 \to v_2 \to$ ...... inductively by choosing v_1 to be any vertex adjacent to v and for each $i > 1$.

Choosing v_{i+1} to be any vertex adjacent to v_i except v_{i-1}, the existence of such a vertex is guaranteed by our hypothesis.

Since G has only finitely many vertices, we must eventually choose a vertex that has been chosen before.

If v_k is the first such vertex, then that part of the walk lying between the two occurrences of v_k is the required cycle.

Theorem 1.19. *A connected graph G is Eulerian if and only if the degree of each vertex of G is even.*

Proof. Suppose that P is an Eulerian trail of G. Whenever P passes through a vertex, there is a contradiction of 2 towards the degree of that vertex.

Since each edge occurs exactly once in P, each vertex must have even degree.

The proof is by induction on the number of edges of G.

Suppose that the degree of each vertex is even.

Since G is connected, each vertex has degree at least 2 and so by lemma, G contains a cycle C.

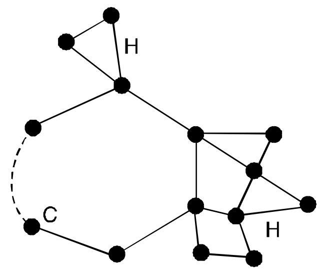
If C contains every edge of G, the proof is complete.

If not, we remove from G the edges of C to form a new, possibly disconnected, graph H with fewer edges that G and in which each vertex still has even degree.

By the induction hypothesis, each component of H has an Eulerian trail.

Since each component of H has at least one vertex in common with C, by connectedness, we obtain the required Eulerian trail of G by following the edges of C until a non-isolated vertex of H is reached, tracing the Eulerian trail of the component of H that contains that vertex, and then continuing along the edges of C until we reach a vertex belonging to another component of H and so on.

The whole process terminates when we return to the initial vertex (see Figure below)



Corollary (1) :

A connected graph is Eulerian if and only if its set of edges can be split up into disjoint cycles.

Corollary (2) :

A connected graph is semi-Eulerian if and only if it has exactly two vertices of odd degree.

Theorem 1.20. *Let G be an Eulerian graph. Then the following construction is always possible, and produces an Eulerian trail of G.*

Start at any vertex u and traverse the edges in an arbitrary manner, subject only to the following results :

(*i*) *erase the edges as they are traversed, and if any isolated vertices result, erase them too ;*

(*ii*) *at each stage, use a bridge only if there is no alternative.*

Proof. We show first that the construction can be carried out at each stage.

Suppose that we have just reached a vertex v.

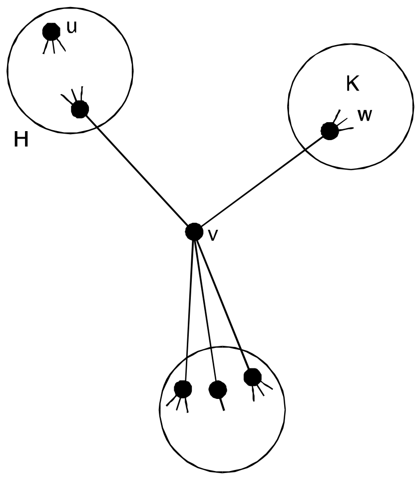
If $v \neq u$ then the subgraph H that remains is connected and contains only two vertices of odd degree u and v.

To show that the construction can be carried out, we must show that the removal of the next edge does not disconnected H or equivalently, that v is incident with atmost one bridge.

But if this is not the case, then there exists a bridge vw such that the component K of H – vw containing w does not contain u (see Figure, below).

Since the vertex w has odd degree in K, some other vertex of K must also have odd degree, giving the required contradiction.

If $v = u$, the proof is almost identical, as long as there are still edges incident with u.



Figure

It remains only to show that this construction always yields a complete Eulerian trail.

But this is clear, since there can be no edges of G remaining untraversed when the last edge incident to u is used, since otherwise the removal of some earlier edge adjacent to one of these edges would have disconnected the graph, contradicting (*ii*).

Theorem 1.21. *(a) If a graph G has more than two vertices of odd degree, then there can be no Euler path in G.*

(b) If G is connected and has exactly two vertices of odd degree, there is an Euler path in G. Any Euler path in G must begin at one vertex of odd degree and end at the other.

Proof. (*a*) Let v_1, v_2, v_3 be vertices of odd degree.

Any possible Euler path must leave (or arrive at) each of v_1, v_2, v_3 with no way to return (or leave) since each of these vertices has odd degree.

One vertex of these three vertices may be the beginning of the Euler path and another the end, but this leaves the third vertex at one end of an untraveled edge.

Thus there is no Euler path.

(*b*) Let u and v be the two vertices of odd degree. Adding the edge $\{u, v\}$ to G produces a connected graph G′ all of whose vertices has even degree. There is an Euler circuit π' in G′.

Omitting $\{u, v\}$ from π' produces an Euler path that begins at u (or v) and ends at v (or u).

Theorem 1.22. *(a) If a graph G has a vertex of odd degree, there can be no Euler circuit in G.*

(b) If G is a connected graph and every vertex has even degree, then there is an Euler circuit in G.

Proof. (*b*) Suppose that there are connected graphs where every vertex has even degree, but there is no Euler circuit. Choose such a G with the smallest number of edges.

G must have more than one vertex since, if there were only one vertex of even degree, there is clearly in Euler circuit. We show first that G must have atleast one circuit. If v is a fixed vertex of G, then since G is connected and has more than one vertex, there must be an edge between v and some other vertex v_1.

This is a simple path (of length 1) and so simple paths exists. Let π_0 be a simple path in G having the longest possible length, and let its vertex sequence be v_1, v_2, ... v_s. Since v_s has even degree and π_0 uses only one edge that has v_s as a vertex, there must be an edge e not in π_0 that also has v_s as a vertex.

If the other vertex of e is not one of the v_i, then we could construct a simple path longer than π_0. Which is a contradiction.

Thus e has some v_i as its other vertex, and therefore we can construct a circuit.

$$v_i, v_{i+1}, \ldots\ldots v_s, v_i \text{ in G.}$$

Since we know that G has circuits, we may choose a circuit π in G that has the longest possible length. Since we assumed that G has no Euler circuits, π cannot contain all the edges of G.

Let G_1 be the graph formed from G by deleting all edges in π (but not vertices).

Since π is a circuit, deleting its edges will reduce the degree of every vertex by 0 or 2, so G_1 is also a graph with all vertices of even degree.

The graph G_1 may not be connected, but we can choose a largest connected component (piece) and call this graph G_2 (G_2 may be G_1).

Now G_2 has fewer edges than G, and so (because of the way G was chosen), G_2 must have an Euler path π'.

If π' passes through all the vertices on G, then π and π' clearly have vertices in common.

If not, then these must be an edge in G between some vertex v' in π', and some vertex v not in π'.

Otherwise we could not get from vertices in π' to the other vertices in G, and G would not be connected.

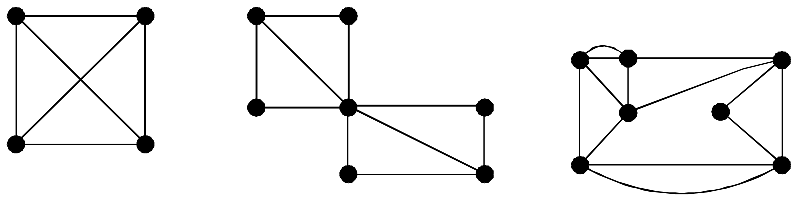
Since e is not in π', it must have been deleted when G_1 was created from G, and so must be an edge in π.

Then v' is also in the vertex sequence of π, and so in any case π and π' have atleast one vertex v' in common. We can then construct a circuit in G that is longer than π by combining π and π' at v'.

This is a contradiction, since π was chosen to be the longest possible circuit in G.

Hence the existence of the graph G always produces a contradiction, and so no such graph is possible.

Problem 1.94. *Which of the graphs in Figure (a), (b), (c) have an Euler circuit, an Euler path but not an Euler circuit, or neither ?*



(a) (b) (c)

Solution. (i) In Figure (a), each of the four vertices has degree 3 ; thus, there is neither an Euler circuit nor an Euler path.

(ii) The graph in Figure (b) has exactly two vertices of odd degree. There is no Euler circuit, but there must be an Euler path.

(iii) In Figure (c), every vertex has even degree ; thus the graph must have an Euler circuit.

1.15 FLEURY'S ALGORITHM

Let G = (V, E) be a connected graph with each vertex of even degree.

Step 1. Select an edge e_1 that is not a bridge in G.

Let its vertices be v_1, v_2.

Let π be specified by $V_\pi : v_1, v_2$ and $E_\pi : e_1$.

Remove e_1 from E and v_1 and v_2 from V to create G_1.

Step 2. Suppose that $V_\pi : v_1, v_2, ... v_k$ and $E_\pi : e_1, e_2, ...e_{k-1}$ have been constructed so far, and that all of these edges and vertices have been removed from v and E to form G_{k-1}.

Since v_k has even degree, and e_{k-1} ends there, there must be an edge e_k in G_{k-1} that also has v_k as a vertex.

If there is more than one such edge, select one that is not a bridge for G_{k-1}.

Denote the vertex of e_k other than v_k by v_{k+1}, and extend V_π and E_π to $V_\pi : v_1, v_2, ..., v_k, v_{k+1}$ and $E_\pi : e_1, e_2, ..., e_{k-1}, e_k$.

Step 3. Repeat step 2 until no edges remain in E.

End of algorithm.

Problem 1.95. *Use Fleury's algorithm to construct an Euler circuit for the graph in Figure (1).*

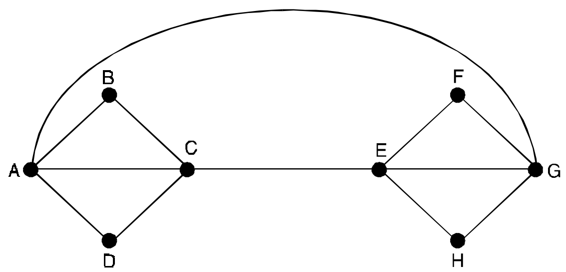


Fig. (1)

Solution. According to step 1, we may begin anywhere.

Arbitrarily choose vertex A. We summarize the results of applying step 2 repeatedly in Table.

Current Path	*Next Edge*	*Reasoning*
π : A	{A, B}	No edge from A is a bridge. Choose any one.
π : A, B	{B, C}	Only one edge from B remains.
π : A, B, C	{C, A}	No edges from C is a bridge. Choose any one.
π : A, B, C, A	{A, D}	No edges from A is a bridge. Choose any one.
π : A, B, C, A, D	{D, C}	Only one edge from D remains.
π : A, B, C, A, D, C	{C, E}	Only one edge from C remains.
π : A, B, C, A, D, C, E	{E, G}	No edge from E is a bridge. Choose any one.
π : A, B, C, A, D, C, E, G	{G, F}	{A, G} is a bridge. Choose {G, F} or {G, H}.
π : A, B, C, A, D, C, E, G, F	{F, E}	Only one edge from F remains.
π : A, B, C, A, D, C, E, G, F, E	{E, H}	Only one edge from E remains.
π : A, B, C, A, D, C, E, G, F, E, H	{H, G}	Only one edge from H remains
π : A, B, C, A, D, C, E, G, F, E, H, G	{G, A}	Only one edge from G remains.
π : A, B, C, A, D, C, E, G, F, E, H, G, A		

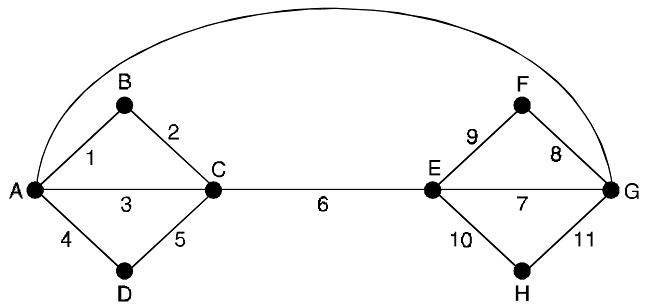


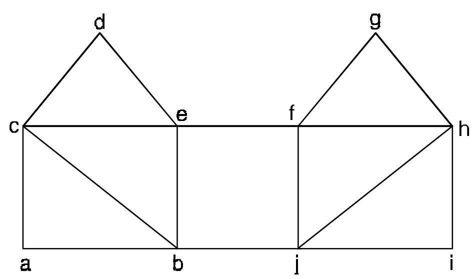
Fig. (2)

The edges in Figure (*b*) have been nembered in the order of their choice in applying step 2.

In several places, other choices could have been made.

In general, if a graph has an Euler circuit, it is likely to have several different Euler circuits.

Problem 1.96. *Using Fleury's algorithm, find Euler circuit in the graph of Figure.*

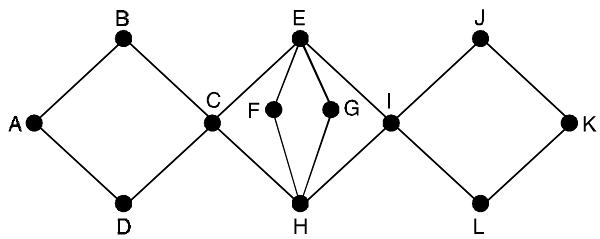


Solution. The degrees of all the vertices are even. There exists an Euler circuit in it.

Current Path	*Next Edge*	*Remark*
$\pi : a$	$\{a, j\}$	No edge from a is a bridge choose (a, j). Add j to π and remove (a, j) from E.
$\pi : aj$	$\{j, f\}$	No edge from j is a bridge. Choose (j, f). Add f to π and remove (j, f) from E.
$\pi : ajf$	$\{f, g\}$	(f, e) is a bridge and (f, g) is not a bridge. Other option (f, h)
$\pi : ajfg$	$\{g, h\}$	(g, h) is the only edge.
$\pi : ajfgh$	$\{h, i\}$	(h, i) is the other option
$\pi : ajfghi$	$\{i, j\}$	(i, j) is the only edge.
$\pi : ajfghij$	$\{j, h\}$	(j, h) is the only edge.
$\pi : ajfghijh$	$\{h, f\}$	(h, f) is the only edge
$\pi : ajfghijhf$	$\{f, e\}$	(f, e) is the only edge
$\pi : ajfghijhfe$	$\{e, d\}$	Other options are (e, c), (e, a)
$\pi : ajfghijhfed$	$\{d, c\}$	(d, c) is the only option.
$\pi : ajfghijhfedc$	$\{c, b\}$	Other options are (c, e), (c, a).
$\pi : ajfghijkfedcb$	$\{b, a\}$	(b, a) is the only option.
$\pi : ajfghijkfedcba$	$\{a, c\}$	Other options are (a, e)
$\pi : ajfghijkfedcbac$	$\{c, e\}$	(c, e) is the only option.
$\pi : ajfghijkfedcbace$	$\{e, a\}$	(e, a) is the only option.
$\pi : ajfghijkfedcbacea$		No edge is remaining in E.

This is the Euler circuit.

Problem 1.97. *Using Fleury's algorithm, find Euler circuit in the graph of Figure.*



Solution. The degree spectrum of the graph is (2, 2, 4, 2, 4, 2, 2, 4, 4, 2, 2, 2) considering the node from A to L in alphabetical order. Since all values are even there exists an Euler circuit in it. The process is summarized in the following table. The start node is A.

S.No.	*Current path*	*Next Edge Considered*	*Remark*
1.	π : A	{A, B}	We select (A, B). Add B to π and remove (A, B) from E.
2.	π : AB	{B, C}	It is the only option. Remove (B, C) from E and B from V. Add C to π.
3.	π : ABC	{C, E}	(C, D) cannot be selected, as it is a bridge. Add E to π and remove (C, E) from E.
4.	π : ABCE	{E, F}	Other options are there.
5.	π : ABCEF	{F, H}	Other option is (H, I). We cannot select
6.	π : ABCEFH	{H, G}	(H, C), as it is a bridge.
7.	π : ABCEFHG	{G, E}	As in Sl. No. 2
8.	π : ABCEFHGE	{E, I}	As in Sl. No. 2
9.	π : ABCEFHGEI	{I, J}	Other options are also there. Edge (I, H) is a bridge.
10.	π : AFCEFHGEIJ	{J, K}	As in Sl. No. 2.
11.	π : ABCEFHGEIJK	{K, L}	As in Sl. No. 2
12.	π : ABCEFHGEIJKL	{L, I}	As in Sl. No. 2
13.	π : ABCEFHGEIJKLI	{I, H}	As in Sl. No. 2
14.	π : ABCEFHGEIJKLIH	{H, C}	As in Sl. No. 2
15.	π : ABCEFHGEIJKLIHC	{C, D}	As in Sl. No. 2
16.	π : ABCEFHGEIJKLIHCD	{D, A}	As in Sl. No. 2
17.	π : ABCEFHGEIJKLIHCDA		This is the Euler cycle

1.16 HAMILTONIAN GRAPHS

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

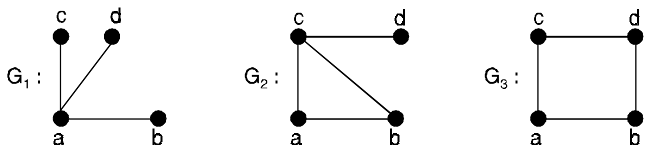
A circuit in a graph G that contains each vertex in G exactly once, except for the starting and ending vertex that **appears twice** is known as **Hamiltonian circuit.**

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

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

Note that an Eulerian circuit traverses every edge exactly once, but may repeat vertices, while a Hamiltonian circuit visists each vertex exactly once but may repeat edges. While there is a criterion for determining whether or not a graph contains an Eulerian circuit, a similar criterion does not exist for Hamiltonian ciruits.

In the following figures, hamiltonian path and cycles are shown :



The graph G_1 has no hamiltonian path (and so hamiltonian cycle), G_2 has hamiltonian path *abcd* but no hamiltonian cycle, while G_3 has hamiltonian cycle *abdca*.

The cycle C_n with n distinct (and n edges) is hamiltonian. Moreover given hamiltonian graph G then if G′ is a subgraph obtained by adding in new edges between vertices of G, G′ will also be hamiltonian. Since any hamiltonian cycle in G will also be hamiltonian cycle in G′. In particular k_n, the complete graph on n vertices, in such a supergraph of a cycle, k_n is hamiltonian.

A simple graph G is called maximal non-hamiltonian if it is not hamiltonian but the addition to it any edge connecting two non-adjacent vertices forms a hamiltonian graph. The graph G_2 is a maximal non-hamiltonian since the addition of an edge *bd* gives hamiltonian graph G_3.

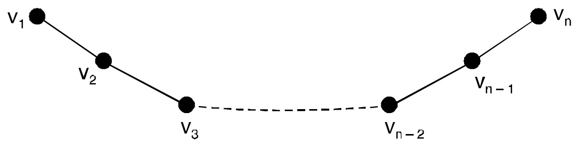
1.17 DIRAC'S THEOREM (1.23)

Let G be a graph of order $p \geq 3$. If deg $v \geq \dfrac{p}{2}$ for every vertex v of G, then G is hamiltonian.

Proof. If $p = 3$, then the condition on G implies that $G \cong k_3$ and hence G is hamiltonian.

We may assume, therefore, that $p \geq 4$.

Let P : $v_1, v_2, \ldots\ldots. v_n$ be a longest path in G (see Figure). Then every neighbour of v_1 and every neighbour of v_n is on P.



Otherwise, there would be a longer path than P.

Consequently, $n \geq 1 + \dfrac{p}{2}$.

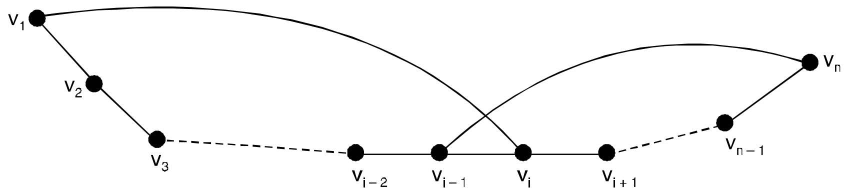
There must be some vertex v_i where $2 \leq i \leq n$, such that v_1 is adjacent to v_i and v_n is adjacent to v_{i-1}.

If this were not the case, then whenever v_1 is adjacent to a vertex v_i, the vertex v_n is not adjacent to v_{i-1}.

Since atleast $\dfrac{p}{2}$ of $p - 1$ vertices different from v_n are not adjacent to v_n.

Hence, deg $v_n \leq (P - 1) - \dfrac{p}{2} < \dfrac{p}{2}$, which contradicts the fact that deg $v_n \geq \dfrac{p}{2}$.

Therefore as we claimed, there must be a vertex v_i adjacent to v_1 and v_{i-1} is adjacent to v_n (see Figure).



We now see that G has cycle C : v_1, v_i, v_{i+1}, ... $v_{n-1}, v_n, v_{i-1}, v_{i-2}$, ..., v_2, v_1 that contains all the vertices of P.

If C contains all the vertices of G (if $n = p$) then C is a hamiltonian cycle, and the proof.

Otherwise, there is some vertex u of G that is not on C.

By hypothesis, deg $u \geq \frac{p}{2}$. Since P contains at least $1 + \frac{p}{2}$ vertices, there are fewer than $\frac{p}{2}$ vertices not on C ; so u must be adjacent to a vertex v that lies on C.

However, the edge uv plus the cycle C contain a path whose length is greater than that of P, which is impossible.

Thus C contains all vertices of G and G is hamiltonian.

Hence the proof.

Corollary :

Let G be a graph with p-vertices. If deg $v \geq \frac{p-1}{2}$ for every vertex v of G then G contains a hamiltonian path.

Proof. If $p = 1$ then G $\cong k_1$ and G contains a (trivial) Hamiltonian path.

Suppose then that $p \geq 2$ and define H = G + k_1.

Let v denote the vertex of H that is not in C.

Since H has vertex $p + 1$, it follows that deg $v \geq p$.

Moreover, for every vertex u of G,

$$\deg_H u = \deg_G u + 1 \geq \frac{p-1}{2} + 1 = \frac{p+1}{2} = \frac{|V(H)|}{2}.$$

By Dirac's theorem, H contains a hamiltonian cycle C. By removing the vertex v from C, we obtain a hamiltonian path in G.

Hence the proof.

Theorem 1.25. *If G is a connected graph of order three or more which is not hamiltonian, then the length k of a longest path of G satisfies $k \geq 2\delta(G)$.*

Proof. Let $p : u_0, u_1$, ..., u_k be a longest path in G.

Since P is longest path, each of u_0 and u_k is adjacent only two vertices of P.

If $u_0u_i \in$ E(G), $1 \leq i \leq k$, then $u_{i-1}\, u_k \notin$ E(G) for otherwise the cycle C : u_0, u_1, u_2, ..., u_{i-1}, u_k, u_{k-1}, u_{k-2}, ..., u_i, u_0 of length $k + 1$ is present in G.

The cycle C cannot contain all vertices of G, since G is not Hamiltonian.

Therefore, there exists a vertex w not on C adjacent with a vertex of C, however this implies G contains a path of length $k + 1$, which is impossible.

Hence for each vertex of $\{u_1, u_2, ..., u_k\}$ adjacent to u_0 there is a vertex of $\{u_0, u_1, ..., u_{k-1}\}$ not adjacent with u_k.

Thus deg $u_k \leq k$ – deg u_0 so that

$$k \geq \deg u_0 + \deg u_k \geq 2\delta(G).$$

Hence the proof.

Theorem 1.26. *Let G be a simple graph with n vertices and let u and v be an edge. Then G is hamiltonian if and only if G + uv is hamiltonian.*

Proof. Suppose G is hamiltonian. Then the super graph G + uv must also be hamiltonian.

Conversely, suppose taht G + uv is hamiltonian.

Then if G is not hamiltonian.

i.e., if G is a graph with $p \geq 3$ vertices such that for all non adjacent vertices u and v, deg u + deg $v \geq p$.

We obtain the inequality deg u + deg $v < n$.

However by hypothesis, deg u + deg $v \geq n$.

Hence G must be hamiltonian.

This completes the proof.

1.18 ORE'S THEOREM (1.27)

If G is a group with $p \geq 3$ vertices such that for all non adjacent vertices u and v, deg u + deg $v \geq p$, then G is hamiltonian.

Proof. Let k denotes the number of vertices of G whose degree does not exceed n,

$$\text{where} \quad 1 \leq n \leq \frac{p}{2}$$

These k vertices induce a subgraph H which is complete, for if any two vertices of H were not adjacent, there would exist two non adjacent vertices, the sum of whose degree is less than p.

This implies that $k \leq n + 1$. However $k \neq n + 1$, for otherwise each vertex of H is adjacent only two vertices of H, and if $u \in$ V(H) and $v \in$ V(G) – V(H), then deg u + deg $v \leq n + (p - n - 2) = p - 2$, which is a contradiction.

Further $k \neq n$; otherwise each vertex of H is adjacent to at most one vertex of G not in H.

However, since $k = n < \frac{p}{2}$, there exists a vertex $w \in$ V(G) – V(H) adjacent to no vertex of H. Then if $u \in$ V(H), deg u + deg $\omega \leq n + (p - n - 1) = p - 1$, which again a contradiction.

Therefore $k < n$, which implies that G satisfies the condition, so that G is Hamiltonian.

Hence the proof.

Problem 1.98. *Let G be a simple graph with n vertices and m edges where m is at least 3. If* $m \geq \frac{1}{2}(n-1)(n-2)+2$.

Prove that G is Hamiltonian. Is the converse true ?

Solution. Let u and v be any two non-adjacent vertices in G.

Let x and y be their respective degrees.

If we delete u, v from G, we get a subgraph with $n-2$ vertices.

If this subgraph has q edges then $q \leq \frac{1}{2}(n-2)(n-3)$.

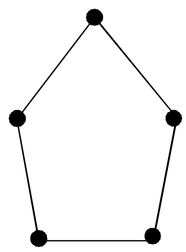
Since u and v are non-adjacent, $m = q + x + y$

Thus, $x + y = m - q \geq \left\{\frac{1}{2}(n-1)(n-2)+2\right\} - \left\{\frac{1}{2}(n-2)(n-3)\right\} = n$.

Therefore, the graph is Hamiltonian.

The converse of the result just proved is not always true.

Because, a 2-regular graph with 5-vertices (see Figure below) is Hamiltonian but the inequality does not hold.



Theorem 1.28. *In a complete graph* K_{2n+1} *there are n edge disjoint Hamiltonian cycles.*

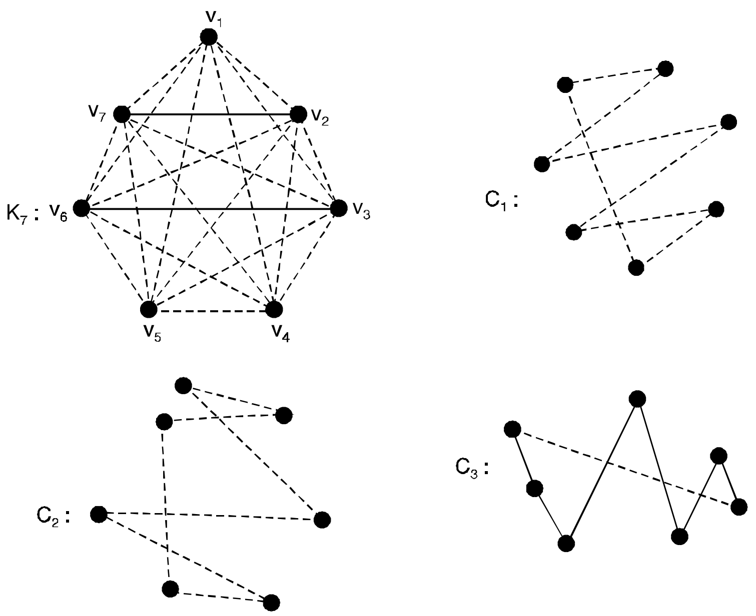
Proof. We first label the vertices of K_{2n+1} as $v_1, v_2, \ldots\ldots v_{2n+1}$ then we construct n paths $P_1, P_2, \ldots\ldots P_n$ on the vertices $v_1, v_2, \ldots\ldots, v_{2n}$ as follows :

$P_i = v_i v_{i-1} v_{i+1} v_{i-2}, \ldots\ldots, v_{i+n-1}, v_{i-n}, \quad 1 \leq i \leq n.$

We note that the jth vertex of P_i is v_k where $k = i + (-1)^{j+1}\left(\frac{j}{2}\right)$, and all subscripts are taken as the integers 1, 2, ..., $2n$ (mod $2n$).

The Hamiltonian cycle C_1 is got by joining v_{2n+1} to the end vertices of P_i.

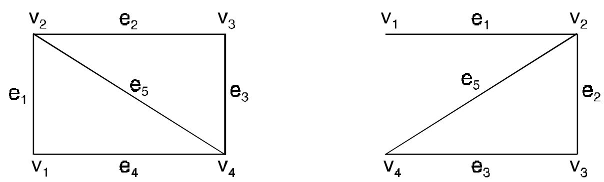
The Figure below illustrates the construction of Hamiltonian cycles in k_7.



It is still an open problem to find a convenient method to determine which graphs are Hamiltonian.

A graph G in which every edge is assigned a real number is called a weighted graph. The real number associated with an edge is called its weight, and the sum of the weights of the edges of G is called the weight of G.

Problem 1.99. *Which of the graphs given in Figure below is Hamiltonian circuit. Give the circuits on the graphs that contain them.*



(a) (b)

Solution. The graph shown in Figure (a) has Hamiltonian circuit given by $v_1e_1v_2e_2v_3e_3v_4e_4v_1$.

Note that all vertices appear in this a circuit but not all edges.

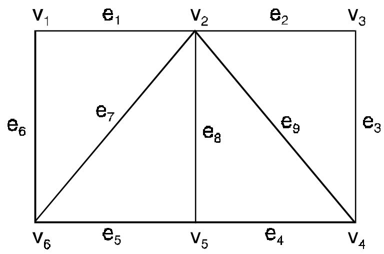
The edge e_5 is not used in the circuit.

The graph shown in Figure (b) does not contain circuit since every circuit containing every vertex must contain the e_1 twice.

But the graph does have a Hamiltonian path $v_1 - v_2 - v_3 - v_4$.

Problem 1.100. *Give an example of a graph which is Hamiltonian but not Eulerian and vice-versa.*

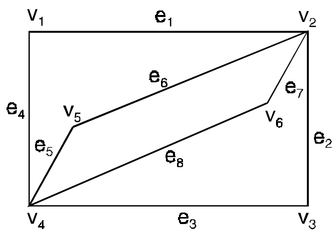
Solution. The following graph shown in Figure below is Hamiltonian but non-Eulerian.



The graph contains a Hamiltonian circuit $v_1e_1v_2e_2v_3e_3v_4e_4v_5e_5v_6e_6v_1$.

Since the degree of each vertex is not n even the graph is non-Eulerian.

The graph shown in Figure below is Eulerian but not Hamiltonian.



The graph is Eulerian since the degree of each vertex is even.

It does not contain Hamiltonian circuit.

This can be seen by noting any circuit containing every vertex must contain a vertex twice except starting vertex and ending vertex.

Problem 1.101. *Show that any k-regular simple graph with 2k – 1 vertices is Hamiltonian.*

Solution. In a k-regular graph, the degree of every vertex is k and $k > k - \frac{1}{2} = \frac{1}{2}(2k - 1) = \frac{n}{2}$.

Where $n = 2k - 1$ is the number of vertices. Therefore, the graph considered is Hamiltonian.

Problem 1.102. *Prove that the complete graph k_n, $n \geq 3$ is a Hamiltonian graph.*

Solution. In k_n,the degree of every vertex is $n - 1$. If $n > 2$, we have $n - 2 > 0$ or $2n - 2 > n$ or $n - 1 > \frac{n}{2}$.

Thus, in k_n, where $n > 2$, the degree of every vertex is greater than $\frac{n}{2}$.

Hence k_n is Hamiltonian.

Theorem 1.29. *Let G be a simple graph on n vertices. If the sum of degrees of each pair of vertices in G is at least n – 1, then there exists a Hamiltonian path in G.*

Proof. We first prove that G is connected.

If not, then G contains at least two components say G_1 and G_2.

Let n_1 and n_2 be the number of vertices of G in the components G_1 and G_2.

Then $n_1 + n_2 \leq n$, the degree of a vertex x of G that is in the component G_1 is atmost $n_1 - 1$ and the degree of a vertex y of G that is in the component G_2 is atmost $n_2 - 1$.

Hence the sum of degrees of the vertices x and y of G is at most $(n_1 + n_2) - 2 \leq n - 2 < n - 1$, a contradiction.

Now we show the existence of the Hamiltonian path, by construction. The construction is as follows :

Step 1. Choose a vertex a of G.

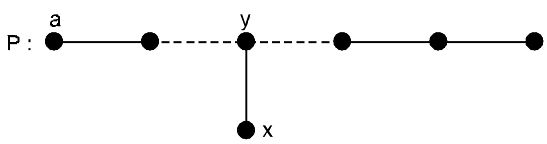
Step 2. Starting from 'a' construct a path P in G.

Step 3. If P is a Hamiltonian path stop, otherwise go to step 4.

Step 4. Extend the path on both the ends to the maximum (make P be a maximal path).

That is if x is a vertex of G adjacent to the end vertex of the path P and not in P, then includes the vertex to P with the corresponding edge and repeats the process. Call the path so obtained as P.

Step 5. If P is a Hamiltonian path then stop. Otherwise, we observe that there exists a vertex x in G that is not in P and adjacent to a vertex y in P (but y is not an end vertex of P).

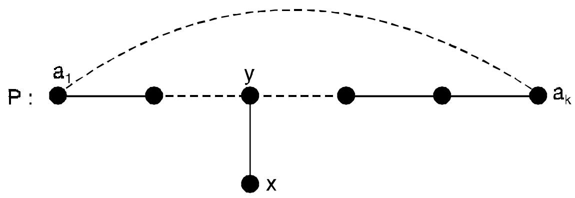


Step 6. Since P is maximal, no vertices of G which are not in P adjacent to the end vertex P.

The end vertices are adjacent to only those vertices in P.

Let P : $a = a_1, a_2, ..., a_k$. Then $k < n$ (otherwise, process stops at step 5).

If a_1 is adjacent to a_k, then obtain a circuit C by join a_1 and a_k, go to step 8. Otherwise, go to step 7 with the following observation.



We observe that there exist i, $1 \leq i \leq k$, such that a_1a_{i+1} and a_ia_k are the edges in G.

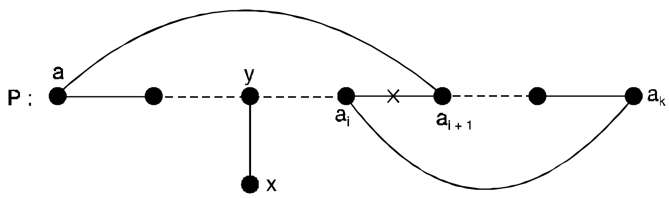
If not, then a_1 is not adjacent to any vertex a_{j+1} in P, which is adjacent to a_k.

But the vertices adjacent to a_k are only the vertices of P (follows by the construction of P), it follows that, if degree of a_k is m, then there are m vertices which are not adjacent to a_1 in P.

Thus, there are at most $k - m - 1$ vertices of P (since a_1 is not adjacent to a_1).

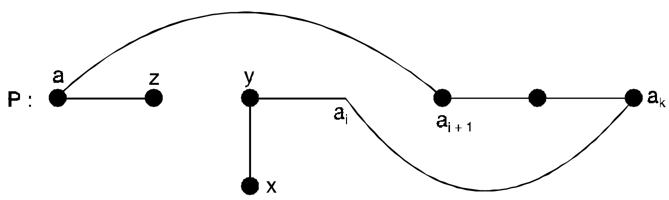
Hence degree of a_1 + degree of $a_k \le (k - m - 1) + m = k - 1 < n - 1$, a contradiction to the assumption made in the statement of the theorem.

Step 7. Construct a circuit C by deleting an edge $a_i a_{i+1}$ in P and joining the edges $a_1 a_{i+1}$ and $a_i a_k$ to P.



Step 8. Join the edge between the vertex x of G and the vertex y in P (the vertices x and y are those vertices which are observed in step 5) to the circuit C. And delete an edge yz incident with y in C.

Step 9. Step 8 yields a path between the vertex x and the vertex z. This path contains one more vertex than the path P so far we have in our hand (*i.e.*, obtained in step 4) call this path as P and go to step 4.



Finally, we note that the process terminates as in each time we are getting a path on one more vertex (that is not in the earlier path) than the earlier path. Moreover, the final output is the desired Hamiltonian path.

Hence the proof.

Theorem 1.30. *In a complete graph with n-vertices there are* $\frac{n-1}{2}$ *edge-disjoint hamiltonian circuits, if n is an odd number* ≥ 3.

Proof : A complete graph with n vertices has $\frac{n(n-1)}{2}$ edges, and a hamiltonian circuit consists of n edges.

Therefore, the number of edge-disjoint hamiltonian circuits in G cannot exceed $\frac{(n-1)}{2}$.

This implies there are $\frac{n-1}{2}$ edge-disjoint hamiltonian circuits, when n is odd it can be shown as by keeping the vertices fixed on a circle, rotate the polygonal pattern clockwise by

$$\frac{360}{(n-1)}, \frac{2.360}{(n-1)}, \frac{3.360}{(n-1)}, ..., \frac{n-3}{2} \cdot \frac{360}{(n-1)} \text{ degrees.}$$

At each rotation we get a hamiltonian circuit that has no edge in common with any of the previous ones. Thus we have $\frac{n-3}{2}$ new hamiltonian circuits, all edges disjoint from one and also edge disjoint among themselves.

Hence the proof.

Problem 1.103. *Which of the simple graphs in Figure have a Hamilton circuit or, if not, a Hamilton path ?*

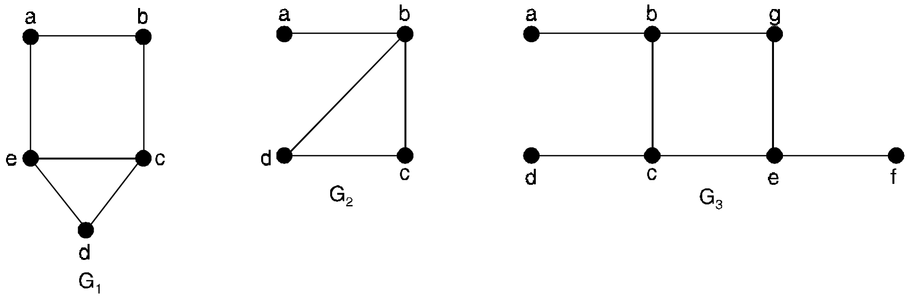


Fig. Three simple graphs.

Solution. G_1 has a Hamilton circuit : *a*, *b*, *c*, *d*, *e*, *a*.

There is no Hamilton circuit in G_2, but G_2 does have a Hamilton path, namely *a*, *b*, *c*, *d*. G_3 has neither a Hamilton circuit nor a Hamilton path, since any path containing all vertices must contain one of the edges $\{a, b\}$ $\{e, f\}$ and $\{c, d\}$ more than once.

Problem 1.104. *Show that neither graph displayed in Figure has a Hamilton circuit.*

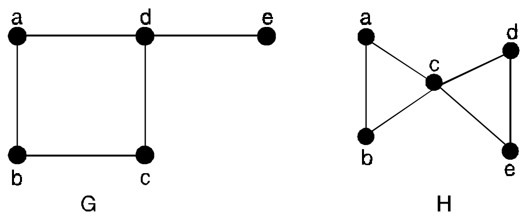
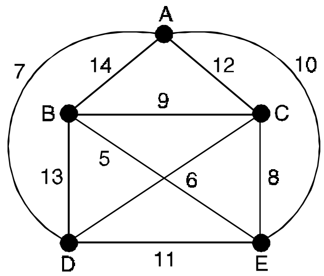


Fig. Two graphs that do not have a Hamilton circuit.

Solution. There is no Hamilton circuit in G since G has a vertex of degree one, namely, *e*. Now consider H. Since the degrees of the vertices *a*, *b*, *d* and *e* are all two, every edge incident with these vertices must be part of any Hamilton circuit. It is now easy to see that no Hamilton circuit can exist in H, for any Hamilton circuit would have to contain four edges incident with C, which is impossible.

Problem 1.105. *Find the minimum Hamiltonian circuit starting from node E in the graph of the Figure.*



Solution. We have to start with the node E. Closest node to E is the node B. Move to B. Now closest node to B is C move to C, extend path up to C and drop node B and all edges from it, from the graph. From C move to D.

From D, move to A and then to E back.

Finally, we have only node E left in the graph.

Thus, we have a Hamiltonian circuit in the graph, which is π : EBCDAE.

The total minimum of this circuit is :

$$EB + BC + CD + DA + EA = 5 + 9 + 6 + 7 + 10 = 37.$$

Problem 1.106. *At a committee meeting of 10 people, every member of the committee has previously sat next to at most four other members. Show that the members may be seated round a circular table in such a way that no one is next to some one they have previously sat beside.*

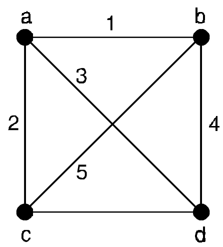
Solution. Consider a graph with 10 vertices representing the 10 members.

Let two vertices be joined by an edge if the corresponding members have not previously sat next to each other.

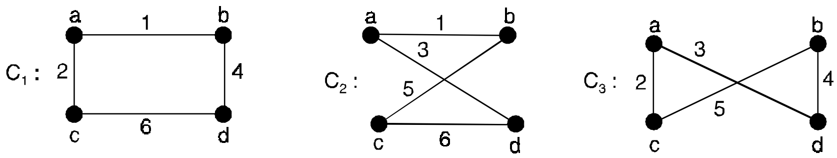
Since any member has not previously sat next to at least five members, the degree of every vertex is at least five.

Therefore, the graph has a Hamiltonian circuit. This circuit provides a seating arrangement of the desired type.

Problem 1.107. *Find three distinct Hamiltonian cycles in the following graph. Also find their weights.*



Solution. The cycles C_1, C_2 and C_3 are three distinct Hamiltonian cycles.



Weight of the cycle $C_1 = 1 + 4 + 6 + 2 = 13$.

Weight of the cycle $C_2 = 1 + 5 + 6 + 3 = 15$

Weight of the third cycle $C_3 = 3 + 4 + 5 + 2 = 14$

Hence the first cycle is of minimum weight.

Theorem 1.31. *A complete graph k_{2n} has a decomposition into n Hamiltonian paths.*

Proof. Consider a complete graph k_{2n}.

Now join a vertex x into K_{2n} and the edges $xv_i \; \forall \; i, \quad 1 \le i \le 2n$.

Then the graph G′ so obtained in K_{2n+1}.

Hence G′ can be decomposed into n Hamiltonian cycles.

Removal of the vertex x from each of these cycles we get n edge disjoint Hamiltonian paths which are the required decomposition of K_{2n}.

Theorem 1.32. *Let G be a connected graph with n vertices, n > 2, and no loops or multiple edges. G has a Hamiltonian circuit if for any two vertices u and v of G that are not adjacent, the degree of u plus the degree of v is greater than or equal to n.*

Corollary : G has a Hamiltonian circuit if each vertex has degree greater than or equal to $\frac{n}{2}$.

Proof. The sum of the degrees of any two vertices is at least $\frac{n}{2} + \frac{n}{2} = n$.

Theorem 1.33. *Let the number of edges of G be m. Then G has a Hamiltonian circuit if* $m \ge \frac{1}{2}(n^2 - 3n + 6)$.

(recall that n is the number of vertices)

Proof. Suppose that u and v are any two vertices of G that are not adjacent. We write deg (u) for the degree of u.

Let H be the graph produced by eliminating u and v from G along with any edges that have u or v as end points.

The H has $n - 2$ vertices and $m - \deg(u) - \deg(v)$ edges (one fewer edge would have been removed if u and v had been adjacent).

The maximum number of edges that H could possibly have is ${}_{n-2}C_2$.

This happens when there is an edge connecting every distinct pair of vertices.

Thus the number of edges of H is at most

$${}_{n-2}C_2 = \frac{(n-2)(n-3)}{2} \quad \text{or} \quad \frac{1}{2}(n^2 - 5n + 6)$$

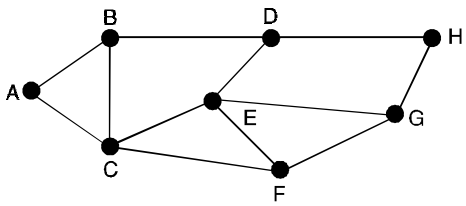
We then have $m - \deg(u) - \deg(v) \leq \frac{1}{2}(n^2 - 5n + 6)$.

Therefore, $\deg(u) + \deg(v) \geq m - \frac{1}{2}(n^2 - 5n + 6)$

By the hypothesis of the theorem,

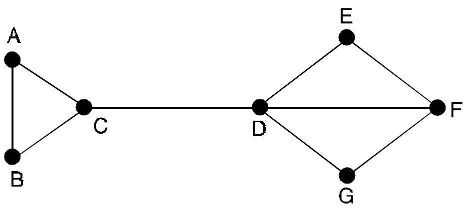
$$\deg(u) + \deg(v) \geq \frac{1}{2}(n^2 - 3n + 6) - \frac{1}{2}(n^2 - 5n + 6) = n.$$

Problem 1.108. *Determine whether a Hamiltonian path or circuit exists in the graph of Figure.*



Solution. Let us take node A to start with. Next, move to either B or C, say B. Extend the path upto B. Next move to D and not to C as a cycle of length 3 could be formed here. Extend the path upto D and drop node B and edges (B, A), (B, C) and (B, D). Then move to H. Drop D and edges from it. Then move to G, then to F (or E) then to E (or F), then to C and finally to A dropping the nodes and edges from them on the way. At the end, only one node A is left with degree zero and π is ABDHGFECA. This is a Hamiltonian cycle.

Problem 1.109. *Determine whether a Hamiltonian path or circuit exists in the graph of Figure.*

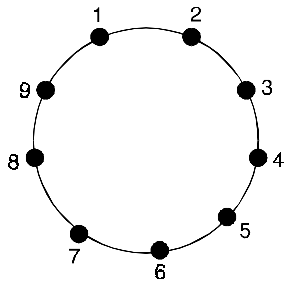


Solution. Let us start with the node A. We can select any one but node C and D. Initialize the path π : A. Next move to the node B we cannot move to C from A. Because any move to D from B and to D from B need node C. Extend the path upto B. Then move to node C, extend the path upto C and drop node B together with edges (B, A) and (B, C). We have now the path π : ABC. Now move to D, extend the path upto D and drop node C together with arcs (C, A) and (C, D). Then move to either node G or E but not to F. Extend the path and do the rest. Finally, proceeding in this way, we get π : ABCDEFG. And two nodes A and G, with degree zero, are left. Thus, this graph has a Hamiltonian path π but no Hamiltonian circuit.

1.19 PROBLEM OF SEATING ARRANGEMENT 1.109

Nine members of a club meet every day for a dinner. They sit in a round table for a dinner, but no two members who sat together will sit together in future. How long is this possible ?

Solution. The seating arrangement can be represented as follows :



Any two numbers can occupy consecutive tables. The neighbouring persons can be represented by an edge. Then each arrangements is a cycle on 9 vertices. These cycles can be chosen from k_9 (since each member can sit with anybody in the beginning).

Thus distinct arrangements as they desired are the edge disjoint (no edges should re-appear, *i.e.,* none of the persons sitting together will sit together in next arrangements) Hamiltonian cycles of K_9, which is possible only for four days (as $9 = 2n + 1 \quad \Rightarrow \quad n = 4$). However this is also possible for 10 members for 4 days only.

If we consider a bench instead of a round table, then for 10 members it is possible for 5 days. (Hamiltonian paths of k_{10}). What can you say about the same situation for nine members.

1.20 TRAVELING-SALESMAN PROBLEM

The traveling-salesman problem, stated as follows :

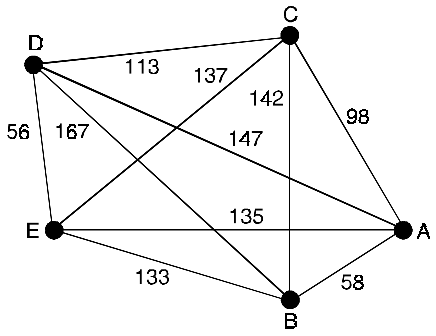
"A salesman is required to visit a number of cities during a trip. Given the distances between the cities, in what order should he travel so as to visit every city precisely once and return home, with the minimum mileage traveled ?" Representing the cities by vertices and the roads between them by edges. We get a graph. In this graph, with every edge e_i there is associated a real number (the distance in miles, say), $w(e_i)$. Such a graph is called a weighted graph ; $w(e_i)$ being the weight of edge e_i.

(*i.e.,* A traveling salesman wants to visit each of n cities exactly once and return to his starting point) if each of the cities has a road to very other city, we have a complete weighted grap.

For example, suppose that the salesman wants to visit five cities, namely, A, B, C, D and E (see Figure). In which order should he visit these cities to travel the minimum total distance ? To solve this problem we can assume the salesman starts in A (since this must be part of the circuit) and examine all possible ways for him to visit the other four cities and then return to A (starting elsewhere will produce the same circuits). There are a total of 24 such circuits, but since we travel the same distance when we travel a circuit in reverse order, we need only consider 12 different circuits to find the minimum total distance he must travel. We list these 12 different circuits and the total distance traveled for each circuit.

As can be seen from the list, the minimum total distance of 458 miles is traveled using the circuit A – B – E – D – C – A (or its reverse).

Route	*Total Distance (miles)*
A – B – D – C – E – A	610
A – B – D – E – C – A	516
A – B – E – C – D – A	588
A – B – E – D – C – A	458
A – B – C – E – D – A	540
A – B – C – D – E – A	504
A – C – B – D – E – A	598
A – C – B – E – D – A	576
A – C – E – B – D – A	682
A – C – D – B – E – A	646
A – D – C – B – E – A	670
A – D – B – C – E – A	728



The graph showing the distance between five cities (A, B, C, D, E)

The traveling salesman problem asks for the circuit of minimum total weight in a weighted, complete, undirected graph that visits each vertex exactly once and returns to its starting point. This is equivalent to asking for a Hamilton circuit with minimum total weight in the complete graph, since each vertex is visited exactly once in the circuit.

The most straight forward way to solve an instance of the traveling salesman problem is to examine all possible Hamilton circuits and select one of minimum total length.

How many circuits do we have to examine to solve the problem if there are n vertices in the graph ? Once a starting point is chosen, there are $(n - 1)$! different Hamilton circuits to examine, since there are $n - 1$ choices for the second vertex, $n - 2$ choices for the third vertex, and so on.

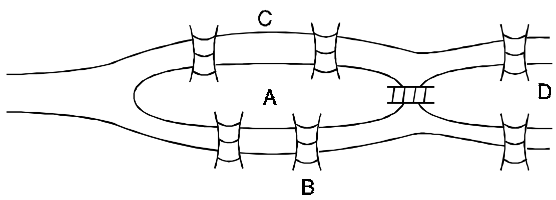
Since a Hamilton circuit can be traveled in reverse order, we need only examine $\frac{(n-1)!}{2}$ circuits to find our answer.

Note that $\dfrac{(n-1)!}{2}$ grows extremely rapidly. Trying to solve a traveling salesman problem in this way when there are only a few dozen vertices is impractical.

For example, with 25 vertices, a total of $\dfrac{24!}{2}$ (approximately 3.1×10^{23}) different Hamilton circuits would have to be considered.

If it took just one nanosecond (10^{-9} second) to examine each Hamilton circuit, a total of approximately ten million year would be required to find a minimum-length Hamilton circuit in this graph by exhaustive search techniques.

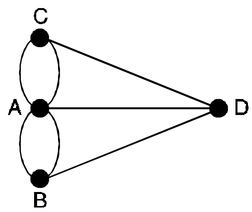
1.21 KÖNIGSBERG'S BRIDGE PROBLEM



There were two islands linked to each other to the bank of the Pregel river by seven bridges as shown above.

The problem was to begin at any of the four land areas, walk across each bridge exactly once and return to the starting point.

One can easily try to solve this problem, but all attempts must be unsuccessful. In proving that, the problem is unsolvable. Euler replaced each land area by a vertex and each bridge by an edge joining the corresponding vertices, there by producing a 'graph' as shown below :



1.22 REPRESENTATION OF GRAPHS

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

Two types of representation are given below :

1.22.1. Matrix representation

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

1.22.2. Adjacency matrix

1.22.2. (*a*) Representation of undirected graph

The adjacency matrix of a graph G with n vertices and no parallel edges is an n by n matrix A = $\{a_{ij}\}$ whose elements are given by $a_{ij} = 1$, if there is an edge between ith and jth vertices, and

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

Note that for a given graph, the adjacency matrix is based on ordering chosen for the vertices.

Hence, there are as many as n ! different adjacency matrices for a graph with n vertices, since there are n ! different ordering of n vertices.

However, any two such adjacency matrices are closely related in that one can be obtained from the other by simply interchanging rows and columns.

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

Observations :

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

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

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

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

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

$$B = A + A^2 + A^3 + ... + A^{n-1}$$

Then G is a connected graph if B has no zero entries of the main diagonal.

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

Adjacency can also be used to represent undirected graphs with loops and multiple edges. A loop at the vertex v_1 is represented by a 1 at the (i, j)th position of the adjacency matrix. When multiple edges are present, the adjacency matrix is no longer a zero-one matrix, since the (i, j)th entry equals the number of edges these are associated to $\{v_i - v_j\}$.

All undirected graphs, including multigraphs and pseudographs, have symmetric adjacency matrices.

1.22.2 (*b*) Representation of directed graph

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

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

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

Observations

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

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

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

(*iv*) The (i, j) entry of A^m is equal to the number of path of length m from vertex v_i to vertex v_j entries of A^T. A shows the in degree of the vertices.

The adjacency matrices can also be used to represent directed multigraphs. Again such matrices are not zero-one matrices when there are multiple edges in the same direction connecting two vertices.

In the adjacency matrix for a directed multigraph a_{ij} equals the number of edges that are associated to (v_i, v_j).

1.22.3. Incidence matrix

1.22.3. (*a*) Representation of undirected graph

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

where $b_{ij} = 1$ when edge e_j is incident with v_i

$= 0$ otherwise

We can make a number of observations about the incidence matrix B of G.

Observations :

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

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

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

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

(*v*) The permutation of any two rows (any two columns) of B corresponds to a labelling of the vertices (edges) of G.

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

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

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

1.22.3. (*b*) Representation of directed graph

The incidence matrix $B = \{b_{ij}\}$ of digraph D with n vertices and m edges is the $n \times m$ matrix in which

$b_{ij} = 1$ if arc j is directed away from a vertex v_i

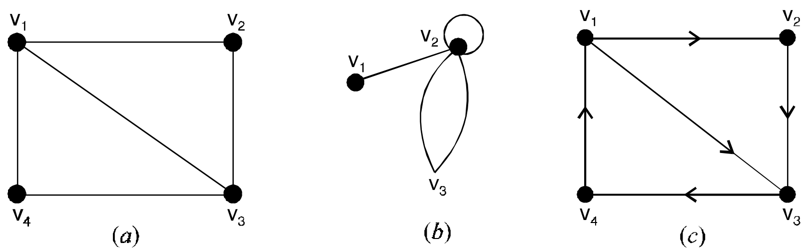
$= -1$ if arc j directed towards vertex v_i

$= 0$ otherwise.

1.22.4. Linked representation

In this representation, a list of vertices adjacent to each vertex is maintained. This representation is also called adjacency structure representation. In case of a directed graph, a case has to be taken, according to the direction of an edge, while placing a vertex in the adjacent structure representation of another vertex.

Problem 1.110. *Use adjacency matrix to represent the graphs shown in Figure below*



Solution. We order the vertices in Figure (1)(a) as v_1, v_2, v_3 and v_4.

Since there are four vertices, the adjacency matrix representing the graph will be a square matrix of order four. The required adjacency matrix A is

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

We order the vertices in Figure (1)(b) as v_1, v_2 and v_3. The adjacency matrix representing the graph is given by

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

Taking the order of the vertices in Figure(1)(c) as v_1, v_2, v_3 and v_4. The adjacency matrix representing the graph is given by

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

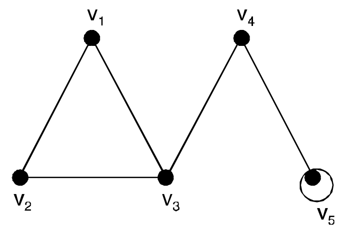
Problem 1.111. *Draw the undirected graph represented by adjacency matrix A given by*

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

Solution. Since the given matrix is a square of order 5, the graph G has five vertices v_1, v_2, v_3, v_4 and v_5.

Draw an edge from v_i to v_j where $a_{ij} = 1$.

The required graph is drawn in Figure below.

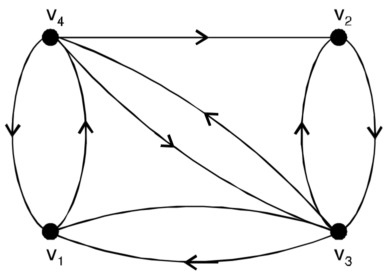


Problem 1.112. *Draw the digraph G corresponding to adjacency matrix*

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

Solution. Since the given matrix is square matrix of order four, the graph G has 4 vertices v_1, v_2, v_3 and v_4. Draw an edge from v_i to v_j where $a_{ij} = 1$.

The required graph is shown in Figure below.



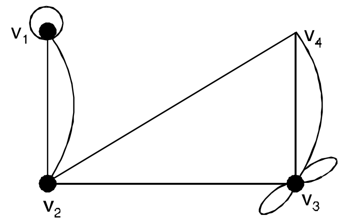
Problem 1.113. *Draw the undirected graph G corresponding to adjacency matrix*

$$A = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 2 & 2 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

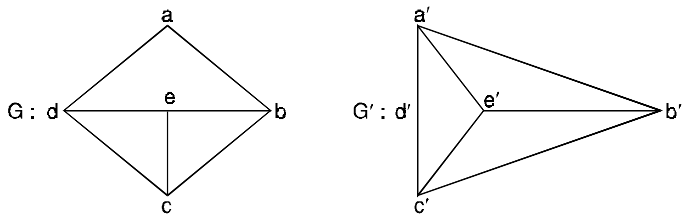
Solution. Since the given adjacency matrix is square matrix of order 4, G has four vertices v_1, v_2, v_3 and v_4. Draw n edges from v_i to v_j where $a_{ij} = n$.

Also draw n loop at v_i where $a_{ij} = n$.

The required graph is shown in Figure below.



Problem 1.114. *Show that the graphs G and G ′ are isomorphic*



Solution. Consider the map $f : G \to G'$ define as $f(a) = d'$, $f(b) = a'$, $f(c) = b'$, $f(d) = c'$ and $f(e) = e'$. The adjacency matrix of G for the ordering a, b, c, d and e is

$$A(G) = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array} \begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

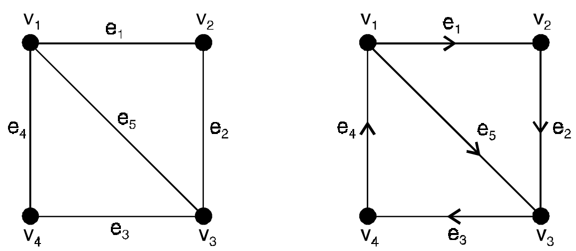
The adjacency matrix of G′ for the ordering d', a', b', c' and e' is

$$A(G') = \begin{array}{c} \\ d' \\ a' \\ b' \\ c' \\ e' \end{array} \begin{array}{c} \begin{array}{ccccc} d' & a' & b' & c' & e' \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

i.e., $A(G) = A(G')$

$\therefore$ G and G′ are isomorphic.

Problem 1.115. *Find the incidence matrix to represent the graph shown in Figure below :*



(a) (b)

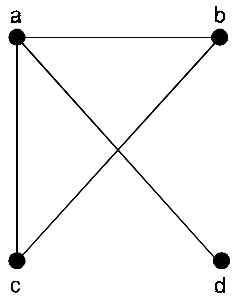
Solution. The incidence matrix of Figure (*a*) is obtained by entering for row *v* and column *e* is 1 if *e* is incident on *v* and 0 otherwise. The incidence matrix is

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

The incidence matrix of the graph of Figure (*b*) is

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

Problem 1.116. *Use an adjacency matrix to represent the graph shown in Figure below :*



Solution. We order the vertices as *a*, *b*, *c*, *d*.

The matrix representing this graph is

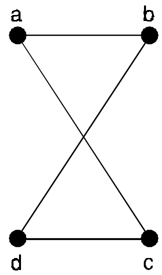
$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Problem 1.117. *Draw a graph with the adjacency matrix*

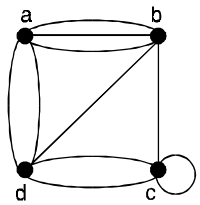
$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

with respect to the ordering of vertices a, b, c, d.

Solution. A graph with this adjacency matrix is shown in Figure below :



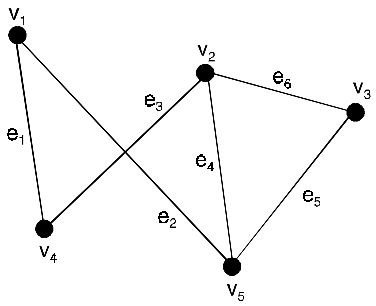
Problem 1.118. *Use an adjacency matrix to represent the pseudograph shown in Figure below :*



Solution. The adjacency matrix using the ordering of vertices *a*, *b*, *c*, *d* is

$$\begin{bmatrix} 0 & 3 & 0 & 2 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}$$

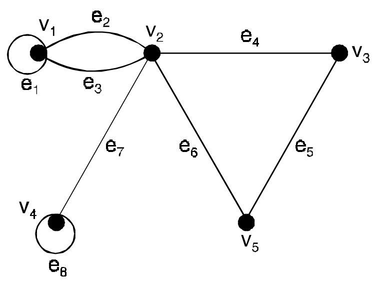
Problem 1.119. *Represent the graph shown in Figure below, with an incidence matrix.*



Solution. The incidence matrix is

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

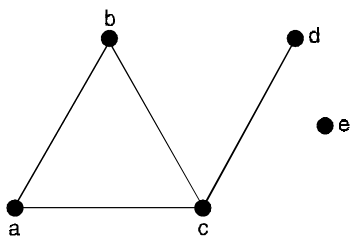
Problem 1.120. *Represent the Pseudograph shown in Figure below, using an incidence matrix.*



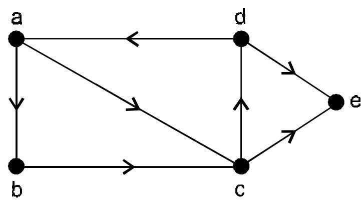
Solution. The incidence matrix for this graph is :

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

Problem 1.121. *Write adjacency structure for the graphs shown in Figure (1)*



(*a*)



(*b*)

Solution. The adjacency structure representation is given in the table for Figure (*a*).
Here the symbol ϕ is used to denote the empty list.

Vertex	*Adjacency list*
a	*b*, *c*
b	*a*, *c*
c	*a*, *b*, *d*
d	*e*
e	ϕ

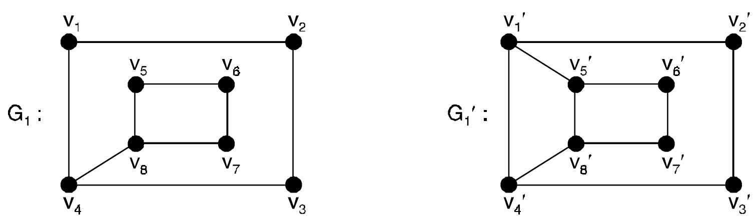
The adjacency structure representation is given in the table for the directed graph shown in Figure (*b*).

Vertex	*Adjacency list*
a	*b*, *c*
b	*c*
c	*d*
d	*a*, *e*
e	*c*

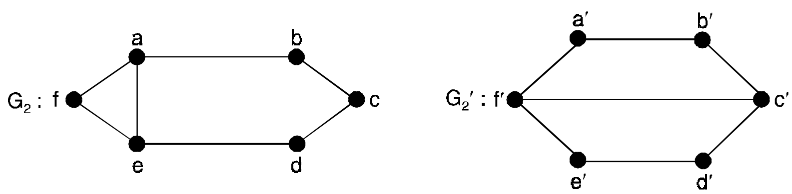
Problem Set 1.1

1. How many vertices do the following graphs have if they contain
   (*i*) 16 edges and all vertices of degree 2
   (*ii*) 21 edges, 3 vertices of degree 4 and others each of degree 3.
2. Suppose a graph has vertices of degree 0, 2, 2, 3 and 9. How many edges does the graph have ?
3. Determine whether the following graphs are isomorphic

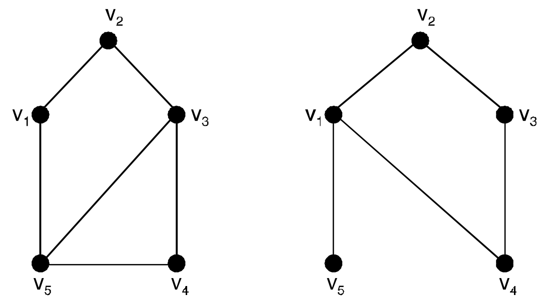
(*i*)



(*ii*)

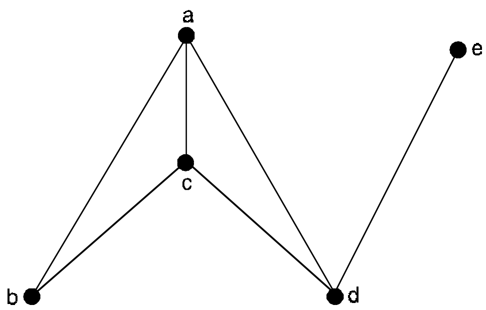


4. Show that graphs are not isomorphic

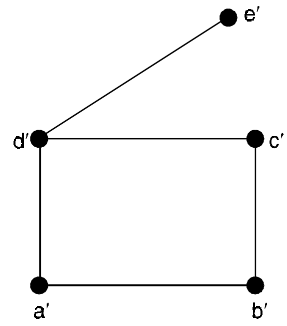


5. Show that the following graphs are isomorphic

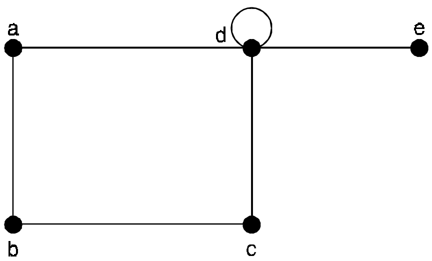
(*i*) G :



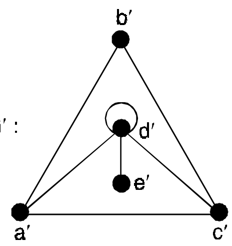
G′ :



(*ii*) G :

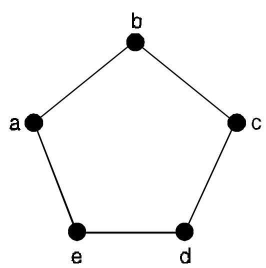


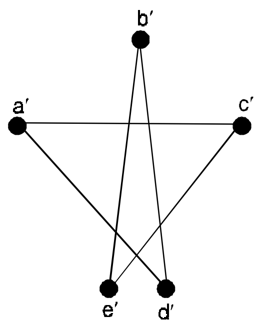
G′ :



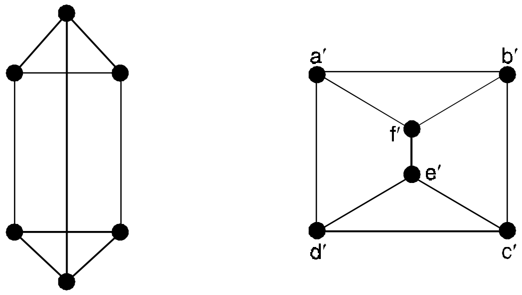
6. Show that the given pairs of graphs are isomorphic

(*i*)



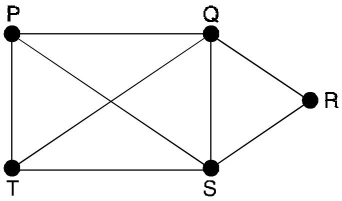


(*ii*)

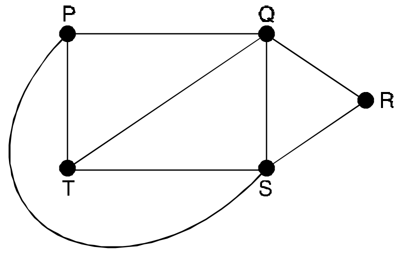


7. Write down the number of vertices, the number of edges, and the degree of each vertex, in

(*i*) the graph in Fig. (*a*) (*ii*) the tree in Fig. (*b*).



(*a*)

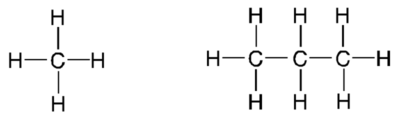


(*b*)

8. Figure below represents the chemical molecules of methane (CH_4) and propane (C_3H_8).

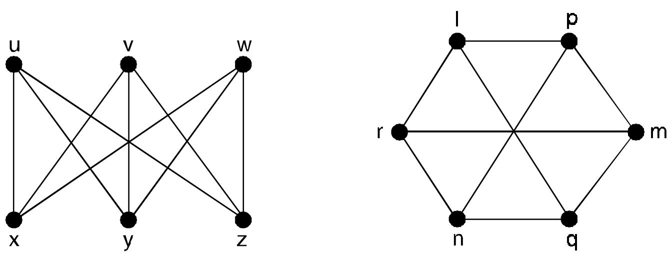
(*i*) Regarding these diagrams as graphs, what can you say about the vertices representing carbon atoms (C) and hydrogen atoms (H) ?

(*ii*) There are two different chemical molecules with formula C_4H_{10}. Draw the graphs corresponding to these molecules.



Methane Propane

9. Write down the vertex set and edge set of each graph in Figure below :



10. Draw (*i*) a simple graph,

(*ii*) a non-simple graph with no loops,

(*iii*) a non-simple graph with no multiple edges, each with five vertices and eight edges.

11. (*i*) Show that there are exactly $2^{n(n-1)/2}$ labelled simple graphs on n vertices

(*ii*) How many of these have exactly m edges ?

12. (*i*) By suitably labelling the vertices, show that the two graphs in Fig. (*a*) are isomorphic

(*ii*) Explain why the two graphs in Fig. (*b*) are not isomorphic.

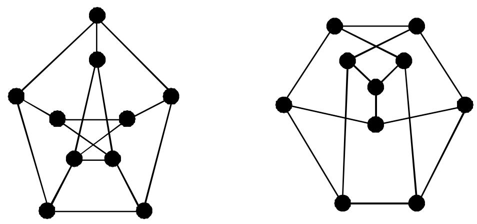


Fig. (*a*)

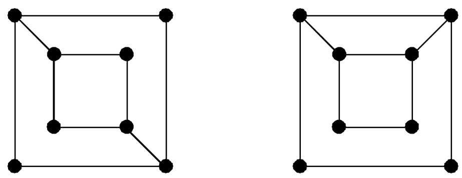
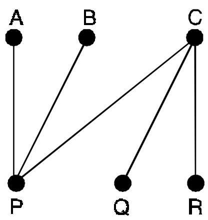


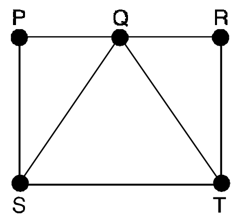
Fig. (*b*)

13. For the graphs shown below, indicate the number of vertices, the number of edges and the degrees of vertices.

(*i*)

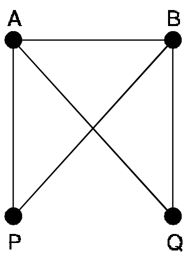


(*ii*)

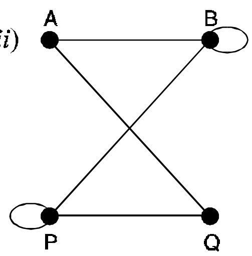


14. Describe the graphs shown below :

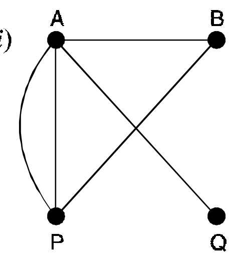
(*i*)



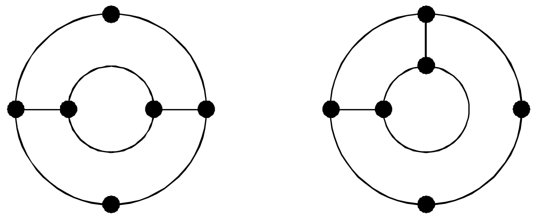
(*ii*)



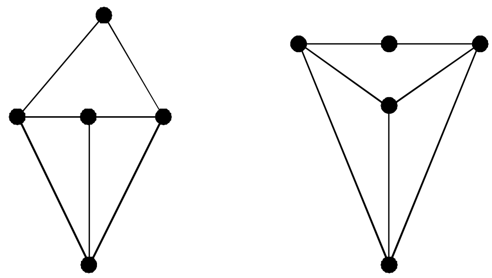
(*iii*)



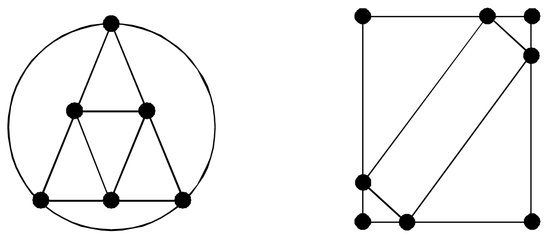
15. Show that the following graphs are not isomorphic :



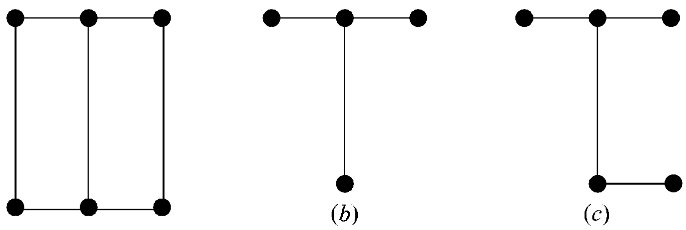
16. Verify that the following graphs are isomorphic :



17. Show that the following graphs are not isomorphic :



18. Three graphs G_1, G_2, G_3 are shown in Figure (*a*), (*b*), (*c*) respectively. Is G_1 a supergraph of G_2 and G_3 ?



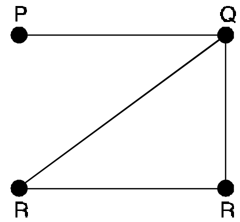
(*a*) (*b*) (*c*)

19. Let G be the graph shown in Figure below. Verify whether H = (V′, E′) is a subgraph of G in the following cases :

(*i*) V′ = {P, Q, S}, E′ = {(P, Q), (P, S)}

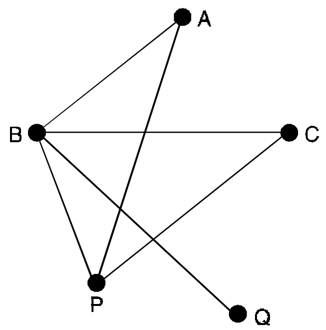
(*ii*) V′ = {Q}, E′ = ϕ, the null set

(*iii*) V′ = {P, Q, R}, E′ = {(P, Q), (Q, R), (Q, S)}



20. For the graph shown in the following Figure, find the nature of the following sequence :

(*i*) BAPCB (*ii*) PABQ (*iii*) CBAPBQ.



21. Prove that the edge set of every closed walk can be partitioned into pairwise edge-disjoint circuits.

22. Show that in a graph with n vertices, the length of a path cannot exceed $n - 1$ and the length of a circuit cannot exceed n.

23. Prove that if u is an odd vertex in a graph G then there must be a path in G from u to another odd vertex v in G.

24. In a graph G, let P_1 and P_2 be two different paths between two given vertices. Prove that G has a circuit in it.

25. Suppose G_1 and G_2 are isomorphic. Prove that if G_1 is connected then G_2 is also connected.

26. Prove that any two simple connected graphs with n vertices, all of degree two, are isomorphic.

27. Show that if G is a connected graph in which every vertex has degree either 1 or 0 then G is either a path or a cycle.

28. Let G be a graph with 15 vertices and 4 components. Prove that atleast one component of G has atleast 4 vertices.

29. If G is a simple graph with n vertices and k components, prove that G has atleast $n - k$ number of edges.

30. Prove that a connected graph of order n contains exactly one circuit if and only if its size is also n.

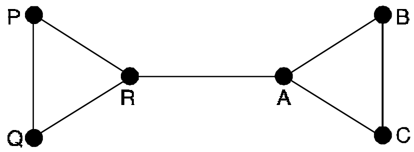
31. Let G be a simple graph. Show that if G is not connected then its complement $\overline{G}$ is connected.

32. Prove that if a connected graph G is decomposed into two subgraphs H_1 and H_2, there must be atleast one vertex common to H_1 and H_2.

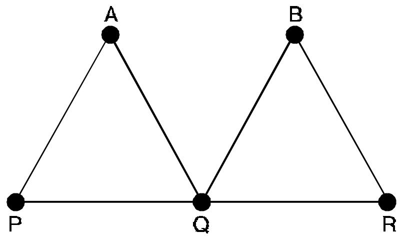
33. Prove that a connected graph is semi-Eulerian if and only if it has exactly zero or two vertices of odd degree.

34. Prove that the Petersen graph is neither Eulerian nor semi-Eulerian.

35. Show that the following graph is not Eulerian :



36. Show that the following graph is Eulerian :

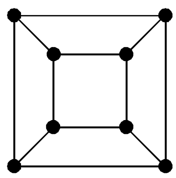
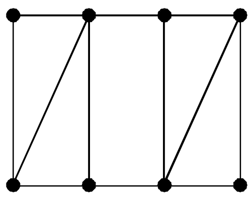


37. Show that the complete graph K_n contains $\frac{1}{2}(n-1)!$ different Hamiltonian circuits.

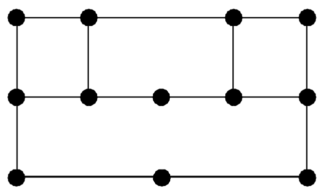
38. Prove that, if G is a bipartite graph with an odd number of vertices then G is non-Hamiltonian.

39. If the degree of each vertex of a simple graph is atleast $\frac{(n-1)}{2}$, where n is the number of vertices, show that the graph has a Hamiltonian path.

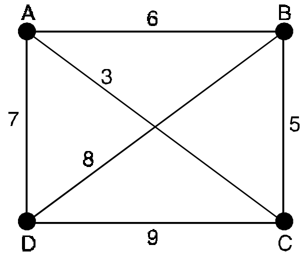
40. Show that the following graphs are Hamiltonian but not Eulerian.

41. Show that the following graph is Hamiltonian.



42. Solve the travelling salesman problem for the weighted graph shown below :



Answers 1.1

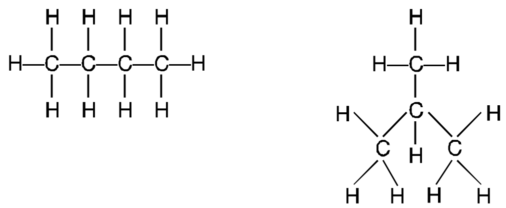
1. (*i*) 16 (*ii*) 13 **2.** 8 **3.** (*i*) Not isomorphic (*ii*) Not isomorphic

7. (*i*) There are 5 vertices and 8 edges ; vertices P and T have degree 3, vertices Q and S have degree 4, and vertex R has degree 2.

(*ii*)There are 6 vertices and 5 edges ; vertices A, B, E and F have degree 1 and vertices C and D have degree 3.

8. (*i*) Each carbon atom vertex has degree 4 and each hydrogen atom vertex has degree 1.

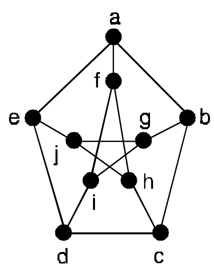
(*ii*) The graphs are as follows :

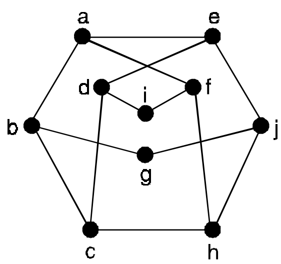


9. V(G) = {*u, v, w, x, y, z*}, E(G) = {*ux, uy, uz, vx, vy, vz, wx, wy, wz*} ;

V(G) = {*l, m, n, p, q, r*}, E(G) = {*lp, lq, lr, mp, mq, mr, np, nq, nr*}.

12. (*i*) We can label the vertices as follows :





(*ii*) In the first graph, no vertices of degree 2 are adjacent, in the second graph they are adjacent in pairs, since isomorphism preserves adjacency of vertices, the graphs are not isomorphic.

13. (*i*) There are 6 vertices and 5 edges ; vertices A, B, Q, R are pendant vertices and vertices C and P have degree 3.

(*ii*) There are 5 vertices and 7 edges ; vertices P and Q have degree 2, S and T have degree 3 and Q has degree 4.

14. (*i*) This is a simple graph with four vertices and five edges. Vertices A and B are of degree 3 and vertices P, Q are of degree 2.

(*ii*) This is a general graph with four vertices and six edges, of which two are self-loops. The vertices A and Q are of degree 2, and B and P are of degree 4.

(*iii*) This is a multigraph with four vertices and five edges. There are parallel edges joining A and P. The degree of A is 4, degree of P is 3, degree of B is 2 and Q is a pendant vertex.

18. Yes **19.** (*i*) No (*ii*) Yes (*iii*) No.

20. (*i*) Circuit (*ii*) Path (*iii*) Open walk which is not a path.

35. Starting with any vertex, it is not possible to return to that vertex without traversing the edge RA twice.

36. The graph contains an Euler line : PAQBRQP.

42. Circuit of least weight : ADBCA ; least total weight 23.

CHAPTER

2

Planar Graphs

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 ?

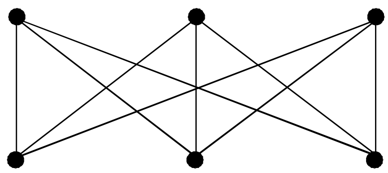


Fig. 2.1. Three houses and three utilities.

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.

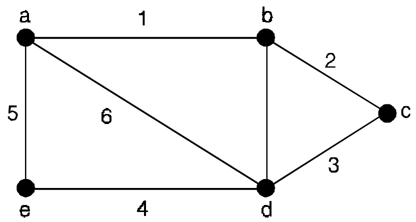
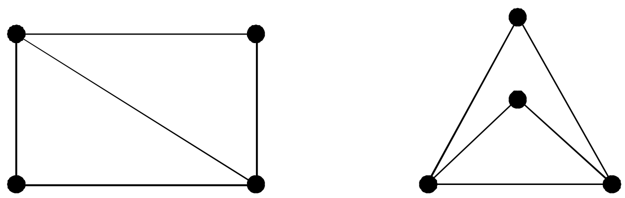


Fig. 2.2. Geometric representation of a graph.

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.



(*a*) (*b*)

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.

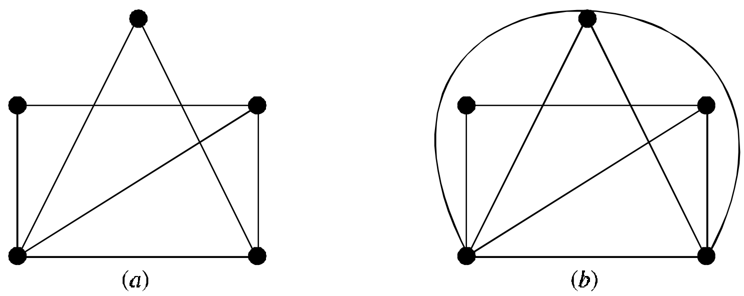


Fig. 2.4.

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.

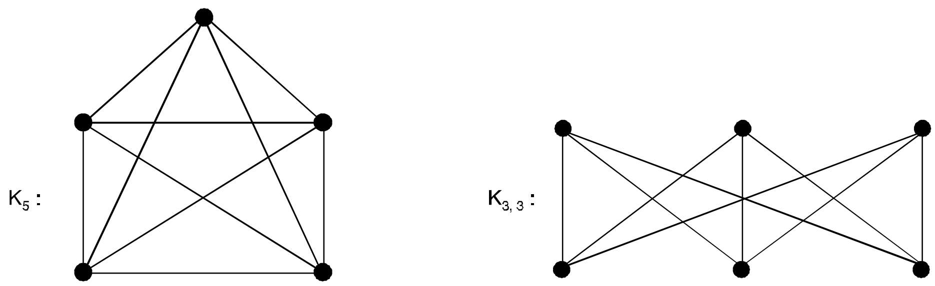


Fig. 2.5.

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.

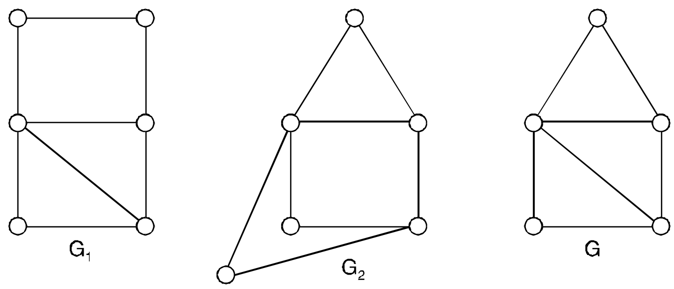


Fig. 2.6. Two homeomorphic graphs obtained from G by adding vertices to edges.

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).

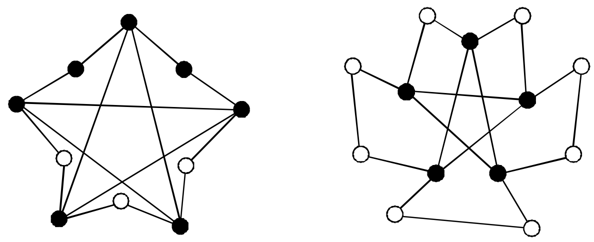


Fig. 2.7. Two homeomorphic graphs obtained from K_5.

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.

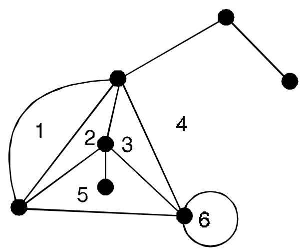


Fig. 2.8. Plane representation (the numbers stand for regions).

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

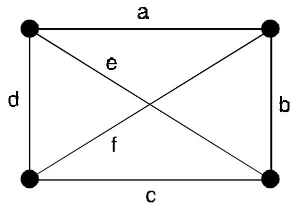


Fig. 2.9.

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.

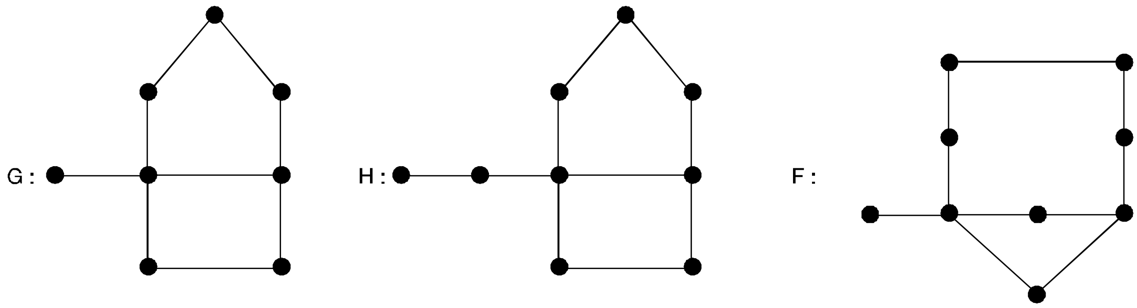


Fig. 2.10.

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.

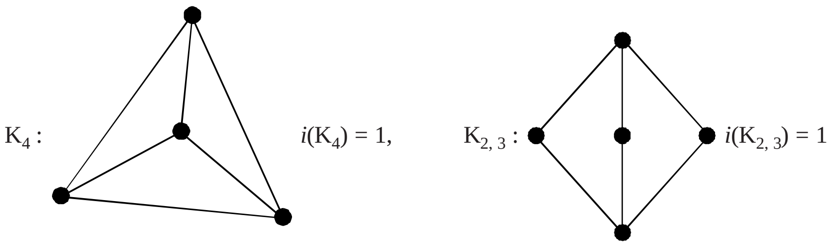


Fig. 2.11.

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,

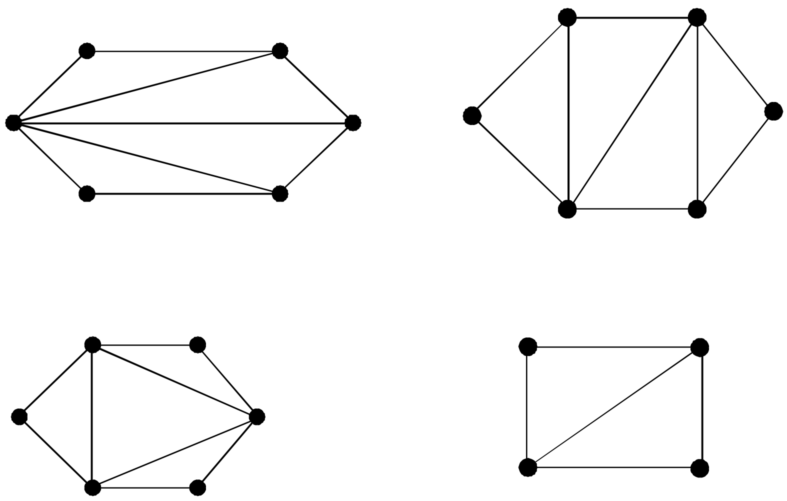
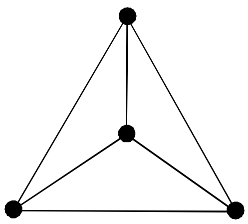


Fig. 2.12. Maximal outer planar graphs.

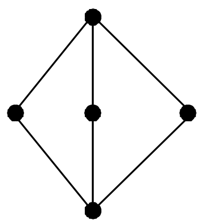
2.9.2. Minimally non-outer planar graphs

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.

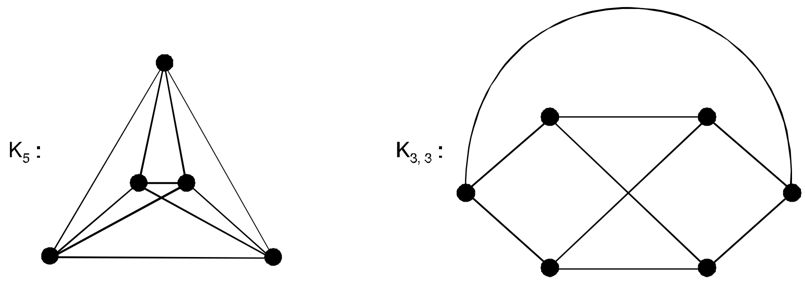


Fig. 2.13. K_5 and $K_{3,\,3}$ are non planar graphs with one crossing.

2.11 BIPARTITE GRAPH

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

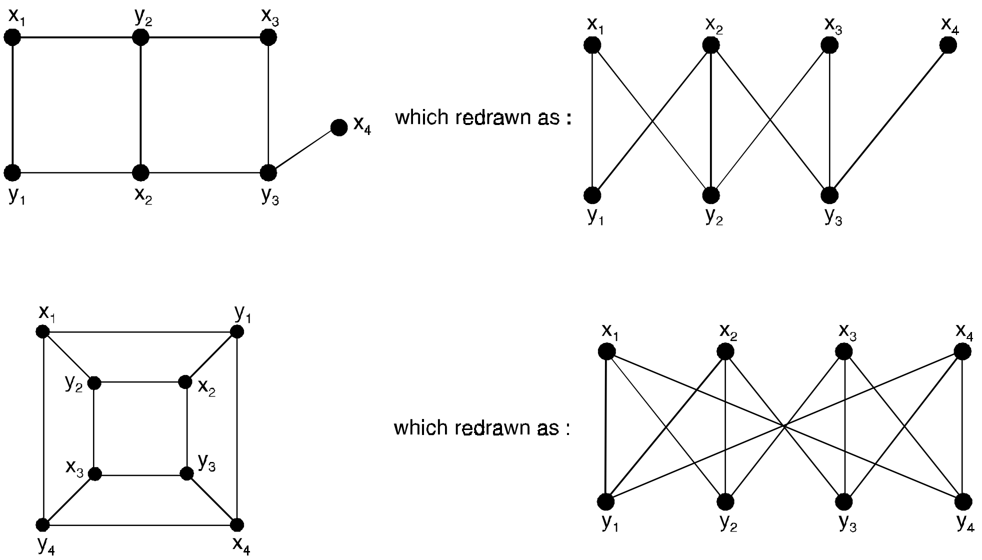


Fig. 2.14. Some bipartite graphs.

2.11.1. Complete bipartite graph

The complete bipartite graph on *m* and *n* vertices, denoted $K_{m,\,n}$ is the graph, whose vertex set is partitioned into sets V_1 with *m* vertices and V_2 with *n* vertices in which there is an edge between each pair of vertices V_1 and V_6. Where V_1 is in V_1 and V_2 is in V_2. The complete bipartite graphs $K_{2,\,3}$, $K_{3,\,3}$, $K_{3,\,5}$ and $K_{2,\,6}$ are shown in Figure below. Note that $K_{r,\,s}$ has $r + s$ vertices and rs edges.

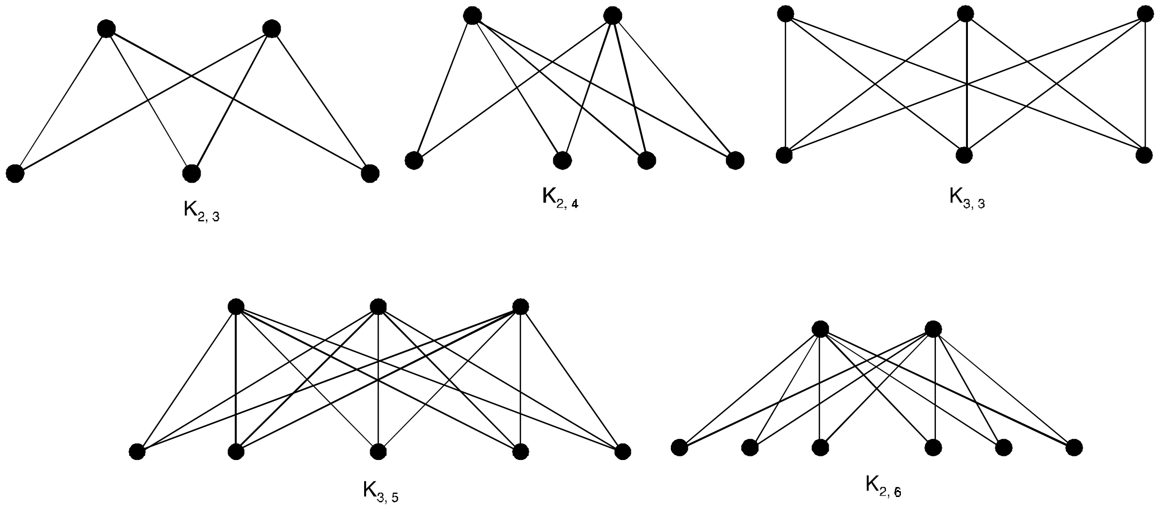


Fig. 2.15. Some complete bipartite graphs.

A complete bipartite graph $K_{m,n}$ is not a regular if $m \neq n$.

Problem 2.1. *Show that C_6 is a bipartite graph.*

Solution. C_6 is a bipartite graph as shown in Figure below.

Since its vertex set can be partitioned into two sets $V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$ and every edge of C_6 connects a vertex in V_1 and a vertex in V_2.

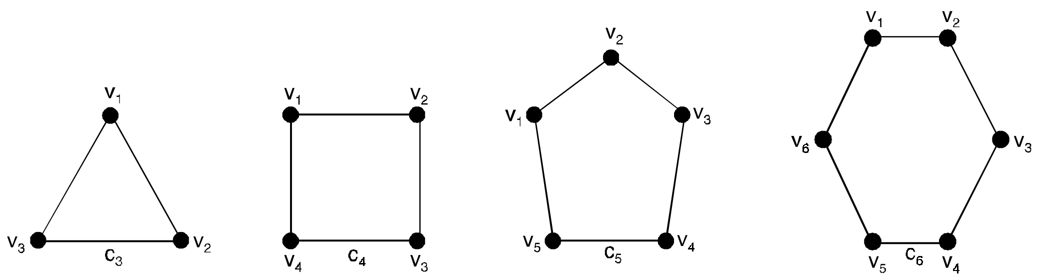
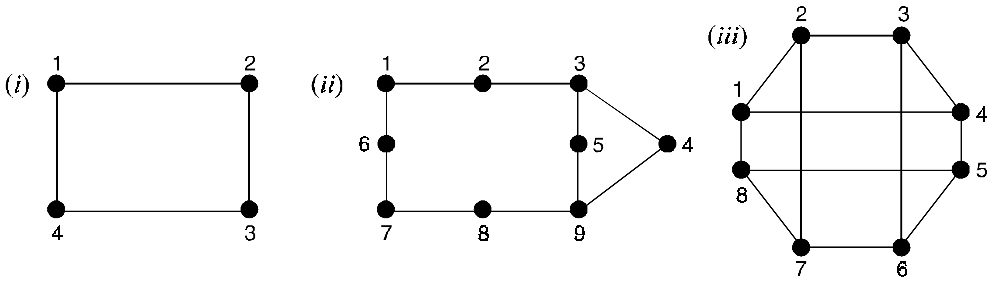


Fig. 2.16.

Problem 2.2. *Prove that a graph which contains a triangle cannot be bipartite.*

Solution. At least two of three vertices must lie in one of the bipartite sets, since these two are joined by two are joined by edge, the graph can not be bipartite.

Problem 2.3. *Determine whether or not each of the graphs is bipartite. In each case, give the bipartition sets or explain why the graph is not bipartite.*



Solution. (*i*) The graph is not bipartite because it contains triangles (in fact two triangles).

(*ii*) This is bipartite and the bipartite sets are {1, 3, 7, 9} and {2, 4, 5, 6, 8}

(*iii*) This is bipartite and the bipartite sets are {1, 3, 5, 7} and {2, 4, 6, 8}.

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 (2.17) below.

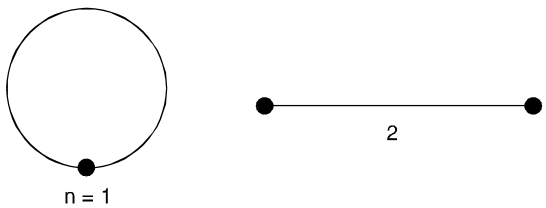


Figure. 2.17. Connected plane graphs with one edge.

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 $n - e + r = \text{K} + 1$.

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 2.4. *Show that the graph K_5 is not coplanar.*

Solution. Since K_5 is a simple graph, the smallest possible length for any cycle K_5 is three.

We shall suppose that the graph is planar.

The graph has 5 vertices and 10 edges so that $n = 5$, $e = 10$.

Now $3n - 6 = 3.5 - 6 = 9 < e$.

Thus the graph violates the inequality $e \leq 3n - 6$ and hence it is not coplanar.

This may be noted that the inequality $e \leq 3n - 6$ is only by a necessary condition but not a sufficient condition for the planarity of a graph.

For example, graph $K_{3,3}$ satisfies the inequality because $e = 9 \leq 3.6 - 6 = 12$, yet the graph is non planar.

Problem 2.5. *Show that the graph $K_{3,3}$ is not coplanar.*

Solution. Since $K_{3,3}$ has no circuits of length 3 (it is bipartite) and has 6 vertices and 9 edges.

i.e., $n = 6$ and $e = 9$ so that $2n - 4 = 2.6 - 4 = 8$.

Hence the inequality $e \leq 2n - 4$ does not satisfy and the graph is not coplanar.

Problem 2.6. *A connected plane graph has 10 vertices each of degree 3. Into how many regions, does a representation of this planar graph split the plane ?*

Solution. Here $n = 10$ and degree of each vertex is 3

$$\Sigma \deg(v) = 3 \times 10 = 30$$

But $\Sigma \deg(v) = 2e \quad \Rightarrow \quad 30 = 2e \quad \Rightarrow \quad e = 15$

By Euler's formula, we have $n - e + r = 2 \quad \Rightarrow \quad 10 - 15 + r = 2 \quad \Rightarrow \quad r = 7$.

Problem 2.7. *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)

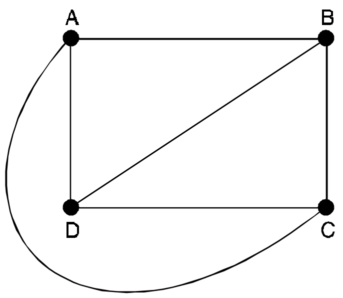


Fig. 2.18.

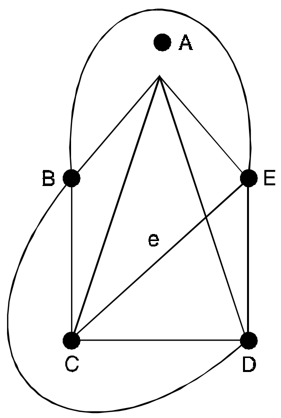


Fig. 2.19.

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

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

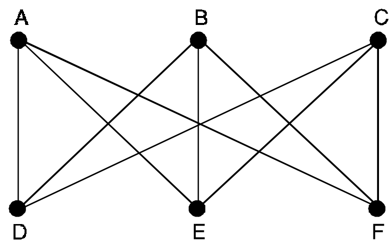


Fig. 2.20.

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, $6 \le 3 * 4 - 6 = 6$

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 $e \le 3v - 6$

i.e., $4 \le 3 * 4 - 6 = 6$ is satisfied.

Hence both K_4 and $K_{2,2}$ are planar.

Problem 2.10. *Determine the number of vertices, the number of edges, and the number of region in the graphs shown below. Then show that your answer satisfy Euler's theorem for connected planar graphs.*

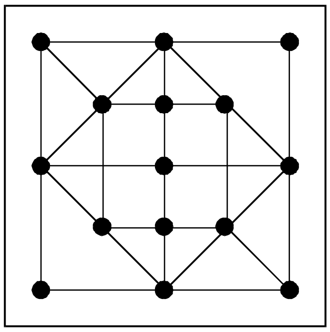


Fig. 2.21.

Solution. There are 17 vertices, 34 edges and 19 regions. So $v - e + r = 17 - 34 + 19 = 2$ which verifies Euler's theorem.

Problem 2.11. *If every region of a simple planar graph with n-vertices and e-edges embeded in a plane is bounded by k-edges then show that* $e = \dfrac{k(n-2)}{k-2}$.

Solution. Since every region is bounded by K-edges, then r-regions are bounded by Kr-edges. Also each edge is counted twice, once for two of its adjacent regions.

Hence we have $2e = \text{K}r \quad \Rightarrow \quad r = \dfrac{2e}{\text{K}}$ ...(1)

i.e., if G is a connected planar graph with n-vertices e-edges and r-regions, then $n - e + r = 2$.

From (1), we have

$$n - e + \frac{2e}{\text{K}} = 2$$

$$\Rightarrow \quad n\text{K} - e\text{K} + 2e = 2\text{K}$$

$$\Rightarrow \quad n\text{K} - 2\text{K} = e\text{K} - 2e$$

$$\Rightarrow \quad \text{K}(n-2) = e(\text{K}-2)$$

$$\Rightarrow \quad e = \frac{\text{K}(n-2)}{\text{K}-2}.$$

Problem 2.12. *Determine whether the graph G shown in Figure (2.22), is planar.*

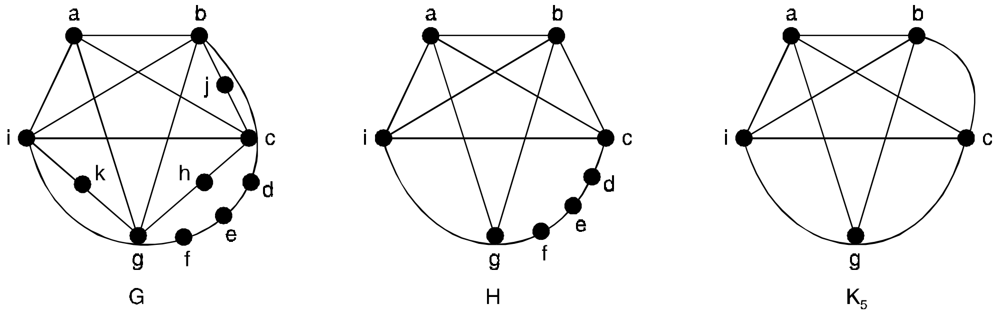


Fig. 2.22. The undirected graph G, a subgraph H homeomorphic to K_5 and K_5.

Solution. G has a subgraph H homeomorphic to K_5, H is obtained by deleting *h, j* and K and all edges incident with these vertices. H is homeomorphic to K_5 since it can be obtained from K_5 (with vertices *a, b, c, g* and *i*) by a sequence of elementary subdivisions, adding the vertices *d*, *e* and *f*.

Hence G is non 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).

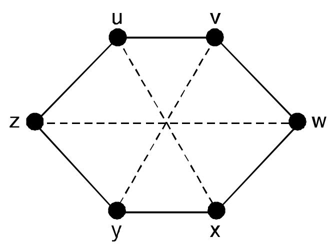


Fig. 2.23.

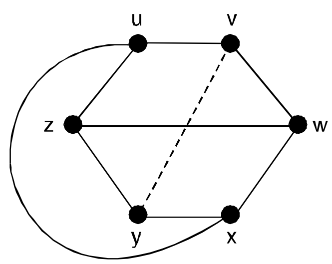


Fig. 2.24.

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).

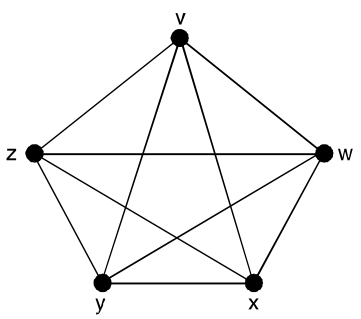


Fig. 2.25.

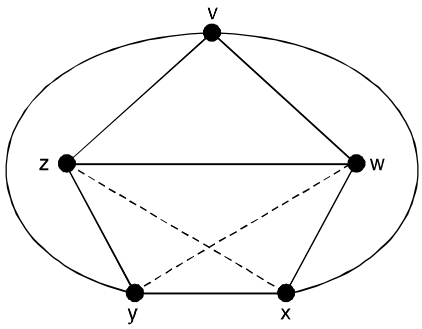


Fig. 2.26.

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* $(k-2)q \leq k(p-2)$.

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., $2q = 2 + 2 + 2 + 3 + 3 + 3 + 4 + 4 + 5 = 28$

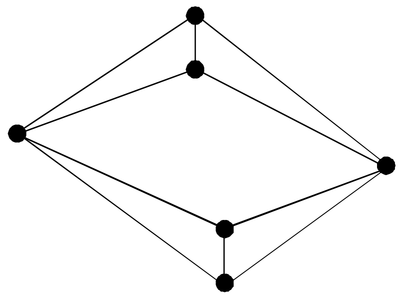
$\Rightarrow \quad q = 24$

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.

Problem 2.15. *Find a graph G with degree sequence (4, 4, 3, 3, 3, 3) such that (i) G is planar (ii) G is non planar.*

Solution. For (*i*) we have drawn a planar graph with six vertices with degree sequence 4, 4, 3, 3, 3, 3 as shown below.



For (*ii*) By Handshaking lemma

$$\sum_{i=1}^{n} \deg v_i = 2q$$

i.e., $2q = 4 + 4 + 3 + 3 + 3 + 3$

$2q = 20$

$\Rightarrow \quad q = 10$

Hence the graph with P = 6, is said to be planar if it satisfies the inequality.

i.e., $q = 3p - 6$

i.e., $10 \leq 3 \times 6 - 6$

or $10 \leq 18 - 6$

$10 \leq 12$

Hence it is not possible to draw a non planar graph with given degree sequence 4, 4, 3, 3, 3, 3.

Problem 2.16. *Determine the number of regions defined by a connected planar graph with 6 vertices and 10 edges. Draw a simple and a non-simple graph.*

Solution. Given $p = 6, q = 10$

Hence by Euler's formula for a planar graph

$$p - q + r = 2$$

$$6 - 10 + r = 2 \quad \Rightarrow \quad r = 6$$

Hence the graph should have 6 regions.

Simple and non-simple graphs with $p = 6$, $q = 10$ and $r = 6$ are shown below.

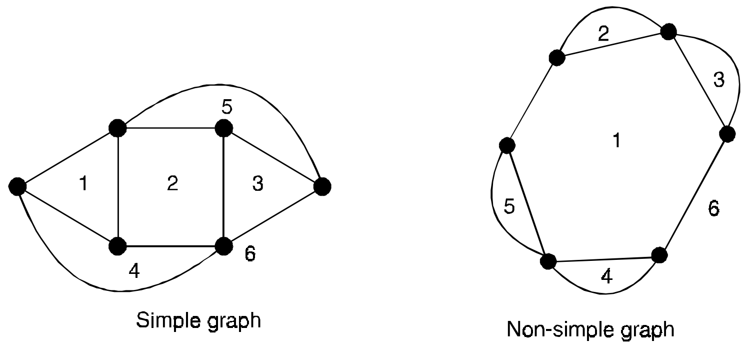
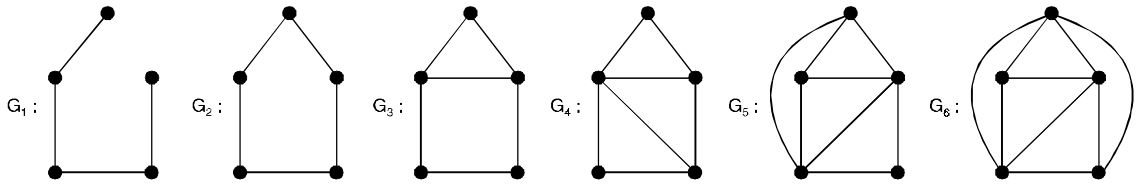


Fig. 2.27.

Problem 2.17. *Draw all planar graphs with five vertices, which are not isomorphic to each other.*

Solution. We have drawn all planar graphs with 5 vertices as shown below.



Problem 2.18. *How many edges must a planar graph have if it has 7 regions and 5 vertices. Draw one such graph.*

Solution. According to Euler's formula, in a planar graph G.

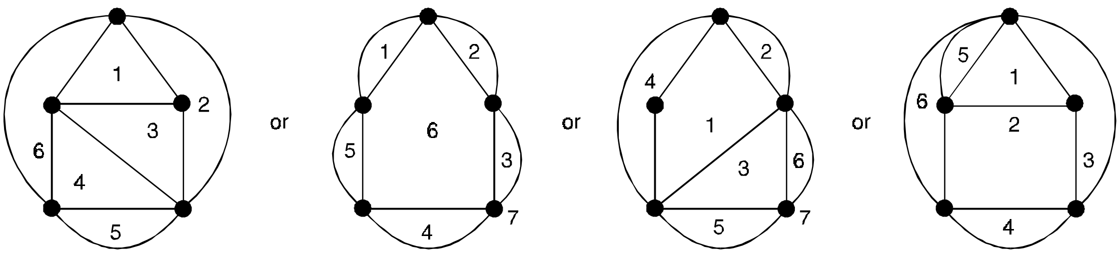
$$p - q + r = 2$$

Here $\quad p = 5, r = 7, q = ?$

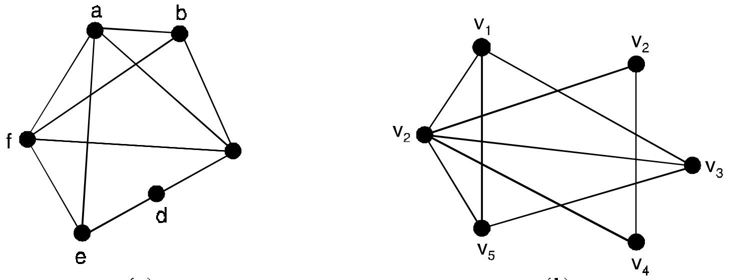
Since the graph is planar, therefore $\quad 5 - q + 7 = 2 \quad \Rightarrow \quad q = 10.$

Hence the given graph must have 10-edges.

Here we have drawn more than one graph as shown below.



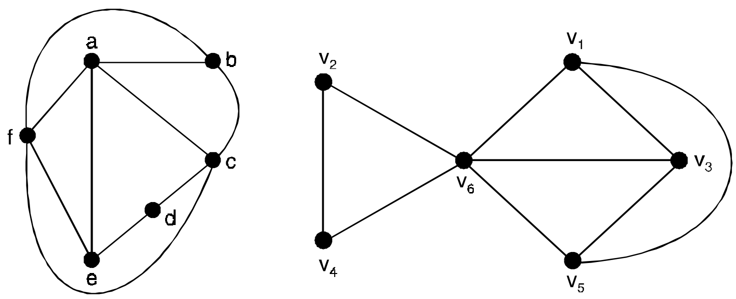
Problem 2.19. *By drawing the graph, show that the following graphs are planar graphs.*



(*a*) (*b*)

Fig. 2.28.

Solution. The graphs shown in Figure (2.28)(*a*, *b*) can be redrawn as planar graphs as follows see Figure (2.29) (*a*, *b*).



(*a*) (*b*)

Fig. 2.29.

Problem 2.20. *Show that the Petersen graph is non planar.*

Solution. Petersen graph is well known non planar graph. Since G has some similarity with K_5 because of 5-cycle, ABCDEA. However since K_5 has vertices of degree 4 only subdivision of K_5 will also have such vertices so G can not have only subdivision of K_5.

Since its vertices each have degree 3. So we look for a subgraph of G which is subdivision of the bipartite graph $K_{3,\,3}$.

The Petersen graph shown in Figure (2.30)(*a*) is non planar since it contains a subgraph homeomorphic to $K_{3,\,3}$ as shown in Figure (2.30)(*c*). Note that the Petersen graph does not contain a subgraph homeomorphic to K_5,

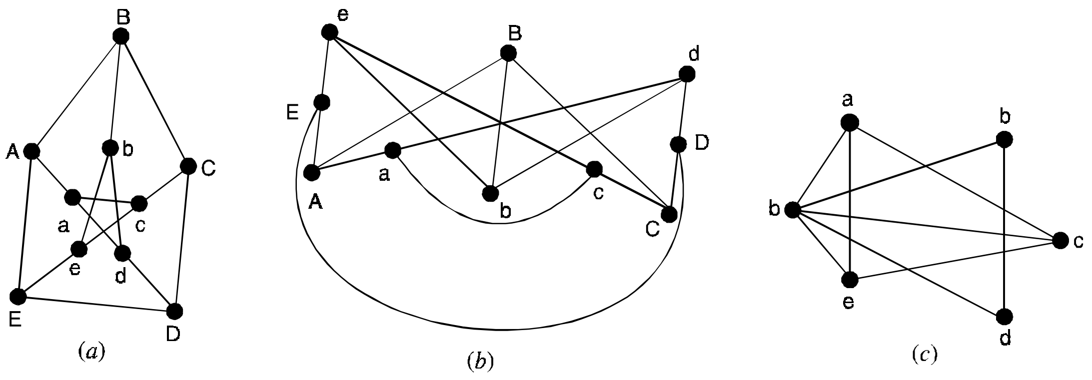


Fig. 2.30.

Problem 2.21. *Find a smallest planar graph that is regular of degree 4.*

Solution. For the graph with two vertices, which is complete, then degree of each vertex is one. For the next smallest graphs are with vertices 3 and 4, if they are complete then degree of each vertex is 2 and 3.

The next graph is with 5 vertices. If degree of each vertex is 4, then it is complete graph with 5 vertices K_5 which is non planar. For the next graph with 6 vertices, if it complete then degree of each vertex is P – 1. *i.e.,* 5. To make this graph 4 regular or regular of degree 4. Remove any 3 non adjacent edges from K_6 we get $K_6 - 3x$ where x is an edge of G, as shown in Figure (2.31), which is regular of degree 4.

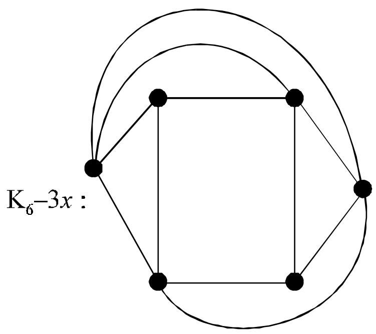
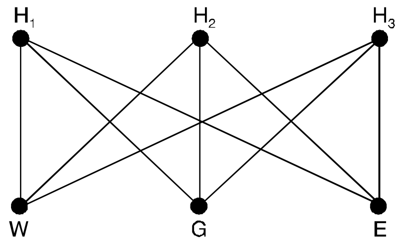


Fig. 2.31.

2.12.1. Three utility problem (2.22)

There are three homes H_1, H_2 and H_3 each to be connected to each of three utilities Water (W), Gas(G) and Electricity(E) by means of conduits. Is it possible to make such connections without any crossovers of the conduits ?

Solution.



The problem can be represented by a graph shown in Figure the conduits are shown as edges while the houses and utility supply centers are vertices.

The above graph is a complete bipartite graph $K_{3,3}$ which is a non planar graph. Hence it is not possible to draw without crossover. Therefore it is not possible to make the connection without any crossover of the conduits.

Problem 2.23. *Is the Petersen graph, shown in Figure below, planar ?*

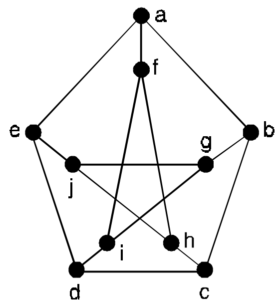


Fig. 2.32. Petersen graph

Solution. The subgraph H of the Petersen graph obtained by deleting *b* and the three edges that have *b* as an end point, shown in Figure (2.33) below, is homeomorphic to $K_{3,3}$ with vertex sets $\{f, d, j\}$

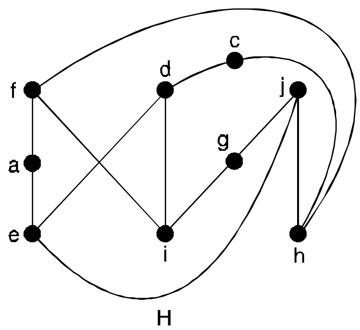


Fig. 2.33.

and $\{e, i, h\}$, since it can be obtained by a sequence of elementary subdivisions, deleting $\{d, h\}$ and adding $\{c, h\}$ and $\{c, d\}$, deleting $\{e, f\}$ and adding $\{a, e\}$ and $\{a, f\}$ and deleting $\{i, j\}$ and adding $\{g, i\}$ and $\{g, j\}$.

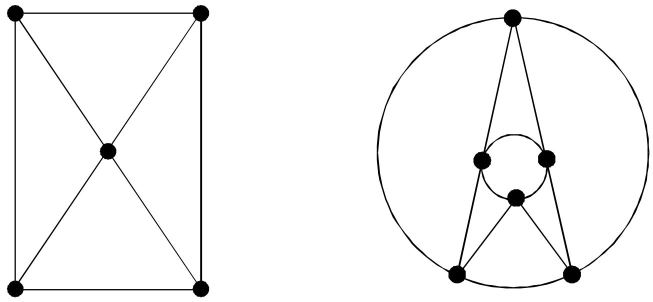
Hence the Petersen graph is not planar.

Problem 2.24. *Show that the following graphs are planar :*

(*i*) *Graph of order 5 and size 8* (*ii*) *Graph of order 6 and size 12.*

Solution. To show that a graph is planar, it is enough if we draw one plane diagram representing the graph in which no two edges cross each other.

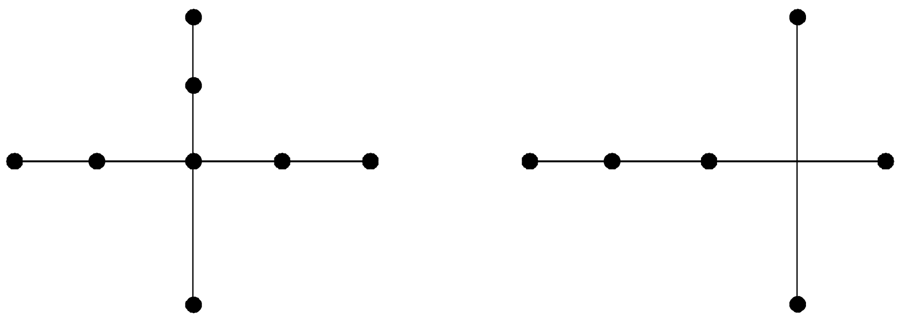
Figure (2.34) (*a*) and (*b*) show that the given graphs are planar.



(*a*) (*b*)

Fig. 2.34.

Problem 2.25. *Verify that the following two graphs are homeomorphic but not isomorphic.*



Solution. Each graph can be obtained from the other by adding or removing appropriate vertices.

Therefore, they are homeomorphic.

That they are not isomorphic is evident if we observe that the incident relationship is not identical.

Problem 2.26. *Show that if a planar graph G of order n and size m has r regions and K components, then* $n - m + r = k + 1$.

Solution. Let H_1, H_2, ... H_k be the K components of G.

Let the number of vertices, the number of edges and the number of non-exterior regions in H_i be n_i, m_i, r_i respectively, $i = 1, 2, \ldots\ldots k$.

The exterior region is the same for all components.

Therefore, $\Sigma n_i = n$, $\Sigma m_i = m$, $\Sigma r_i = r - 1$

If the exterior region is not considered, then the Euler's formula applied to H_i yields

$$n_i - m_i + r_i = 1$$

On summation (from $i = 1$ to $i = k$) this yields

$$n - m + (r - 1) = k$$

$$\Rightarrow \quad n - m + r = k + 1$$

Problem 2.27. *Let G be a connected simple planar (n, m) graph in which every region is bounded by at least k edges. Show that* $m \leq \dfrac{k(n-2)}{(k-2)}$.

Solution. Since every region in G is bounded by at least k edges, we have $2m \geq kr$ ...(1)

Where r is the number of regions

Substituting for r from the Euler's formula in (1), we get

$$2m \geq k(m - n + 2)$$

$$\Rightarrow \quad k(n - 2) \geq km - 2m$$

$$\Rightarrow \quad m \leq \frac{k(n-2)}{(k-2)}$$

Problem 2.28. *Let G be a simple connected planar graph with fewer than 12 regions, in which each vertex has degree at least 3. Prove that G has a region bounded by at most four edges.*

Solution. Suppose every region in G bounded by at least 5 edges.

Then, if G has n vertices and m edges,

we have, $\quad 2m \geq 5r$ ...(1)

Since each vertex has degree atleast 3, the sum of the degrees of the vertices is greater than or equal to $3n$. By virtue of the handshaking property, this means that

$$2m \geq 3n \qquad \text{...(2)}$$

By Euler's formula, we have

$$r = m - n + 2$$

$$\geq m - \left(\frac{2}{3}\right)m + 2 \qquad (\because \quad (2))$$

$$= \frac{m}{3} + 2 \geq \frac{5}{6}r + 2 \qquad (\because \quad (1))$$

This yields $6r \geq 5r + 12$, $r \geq 12$.

This is a contradiction, because G has fewer than 12 regions.

Hence, some region in G is bounded by atmost four edges.

Problem 2.29. *Show that these does not exist a connected simple planar graph with* $m = 7$ *edges and with degree* $\delta = 3$.

Solution. Suppose there is a graph G of the desired type.

Then, for this graph, the inequality $\delta \leq \left(\dfrac{2m}{n}\right)$ gives $3n \leq 14$.

On the other hand, $7 \leq 3n - 6$ or $3n \geq 13$.

Thus, we have $13 \leq 3n \leq 14$ which is not possible (because n has to be a positive integer).

Hence the graph of the desired type does not exist.

Problem 2.30. *Show that every simple connected planar graph G with less than 12 vertices must have a vertex of degree ≤ 4.*

Solution. Suppose every vertex of G has degree greater than or equal to 5.

Then, if $d_1, d_2, d_3, ... d_n$ are the degrees of n vertices of G, we have $d_1 \geq 5$, $d_2 \geq 5$, ... $d_n \geq 5$.

So that $d_1 + d_2 + ... + d_n \geq 5n$.

or $2m \geq 5n$, by handshaking property,

or
$$\frac{5n}{2} \leq m \qquad ...(1)$$

On the other hand, $m \leq 3n - 6$

Thus, we have, in view of (1)

$$\frac{5n}{2} \leq 3n - 6 \quad \text{or} \quad n \geq 12.$$

Thus, if every vertex of G has degree ≥ 5, then G must have at least 12 vertices.

Hence, if G has less than 12 vertices, it must have a vertex of degree < 5.

Problem 2.31. *Show that the condition $m \leq 3n - 6$ is not a sufficient condition for a connected simple graph with n vertices and m edges to be planar.*

Solution. Consider the graph $K_{3,3}$ which is simple and connected and which has $n = 6$ vertices and $m = 9$ edges.

We check that, for this graph, $m \leq 3n - 6$.

But the graph is non-planar.

Problem 2.32. *What is the minimum number of vertices necessary for a simple connected graph with 11 edges to be planar ?*

Solution. For a simple connected planar (n, m) graph,

We have, $m \leq 3n - 6$

or
$$n \geq \frac{1}{3}(m + 6)$$

When $m = 11$, we get $n \geq \frac{17}{3}$.

Thus, the required minimum number of vertices is 6.

Problem 2.33. *Verify Euler's formula for the graph shown in Figure (2.35).*

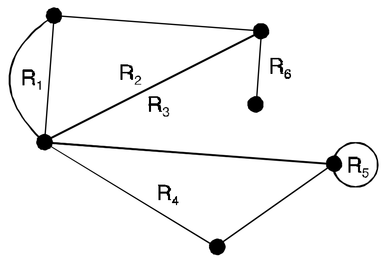


Fig. 2.35.

Solution. The graph has $n = 6$ vertices, $m = 10$ edges and $r = 6$ regions.

Therefore $n - m + r = 6 - 10 + 6 = 2$

Thus, Euler's formula is verified.

Problem 2.34. *What is the maximum number of edges possible in a simple connected planar graph with eight vertices ?*

Solution. When $n = 8$,

$$m \leq 3n - 6 = 18$$

Thus, the maximum number of edges possible is 18.

Theorem 2.5. *A graph is planar if and only if each of its blocks is planar.*

Theorem 2.6. *Every 2-connected plane graph can be embedded in the plane so that any specified face is the exterior.*

Proof. Let f be a non exterior face of a plane block G. Embed G on a sphere and call some point interior to f the North pole.

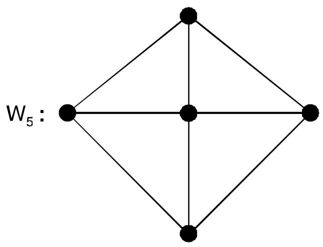
Consider a plane tangent to the sphere at the South pole and project G onto that plane from the North pole.

The result is a plane gr aph isomorphic to G in which f is the exterior face.

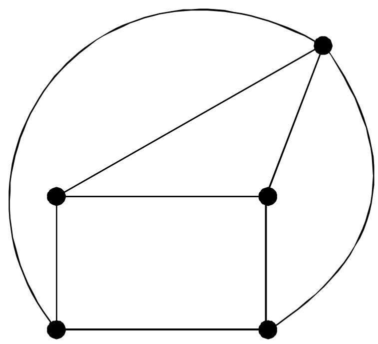
Corollary :

Every planar graph can be embedded in the plane so that a prescribed line is an edge of the exterior region.

Theorem 2.7. *Every maximal planar graph with P ≥ 4 points is 3-connected.*



(*a*)



(*b*)

Fig. 2.36. Plane wheels.

There are five ways of embedding the 3-connected wheel W_5 in the plane : one looks like Figure (2.36)(*a*) and the other four look like Figure (2.36)(*b*).

However, there is only one way of embedding W_5 on a sphere, an observation which holds for all 3-connected graphs.

Theorem 2.8. *Every 3-connected planar graph is uniquely embeddable on the sphere.*

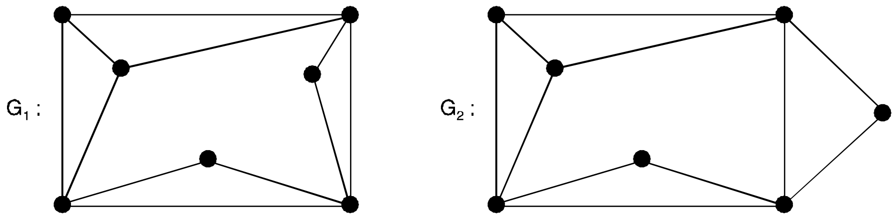


Fig. 2.37. Two plane embeddings of a 2-connected graph.

To show the necessity of 3-connectedness, consider the isomorphic graphs G_1 and G_2 of connectivity 2 shown in Figure above.

The graph G_1 is embedded on the sphere so that none of its regions are bounded by five edges while G_2 has two regions bounded by five edges.

Theorem 2.9. *A graph is the 1-skeleton of a convex 3-dimensional polyhedron if and only if it is planar and 3-connected.*

Theorem 2.10. *Every planar graph is isomorphic with a plane graph in which all edges are straight segments.*

Theorem 2.11. *A graph G is outer planar if and only if each of its blocks is outerplanar.*

Theorem 2.12. *Let G be a maximal outerplane graph with P $\geq$ 3 vertices all lying on the exterior face. Then G has P – 2 interior faces.*

Proof. Obviously the result holds for P = 3.

Suppose it is true for P = n and let G have P = n + 1 vertices and m interior faces.

Clearly G must have a vertex v of degree 2 on its exterior face.

In forming G – v we reduce the number of interior faces by 1 so that m – 1 = n – 2.

Thus m = n – 1 = P – 2, the number of interior faces of G.

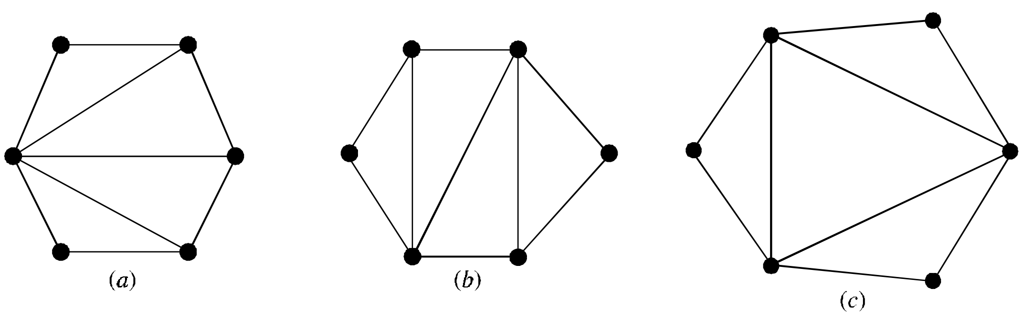


Fig. 2.38. Three maximal outerplanar graphs.

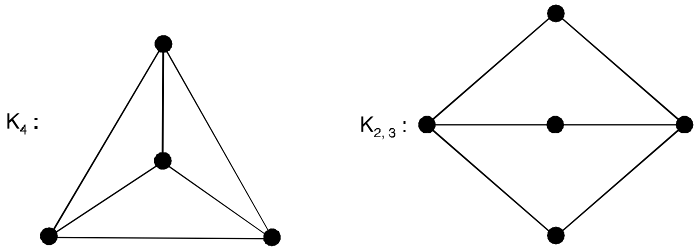


Fig. 2.39. The forbidden graphs for outer planarity.

Corollary :

Every maximal out planar graph G with P points has

(*a*) 2P – 3 lines

(*b*) at least three points of degree not exceeding 3.

(*c*) at least two points of degree 2.

(*d*) K(G) = 2.

All plane embeddings of K_4 and $K_{2,3}$ are of the forms shown in Figure (2.39) above, in which each has a vertex inside the exterior cycle.

Therefore, neither of these graphs is outer planar.

Theorem 2.13. *A graph is outer planar if and only if it has no subgraph homeomorphic to K_4 or $K_{2,3}$ except $K_4 - x$.*

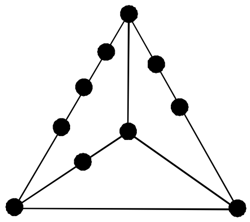


Fig. 2.40. A homeomorph of K_4.

Theorem 2.14. *Every planar graph with atleast nine points has a non planar complement, and nine is the smallest such number.*

Theorem 2.15. *Every outerplanar graph with atleast seven points has a non outer planar complement, and seven is the smallest such number.*

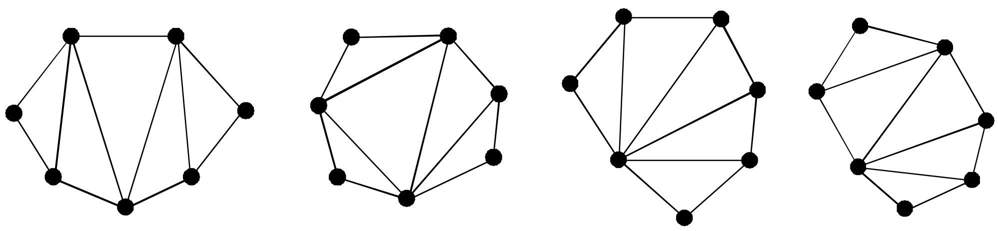


Fig. 2.41. The four maximal outer planar graphs with seven points.

Proof. To prove the first part, it is sufficient to verify that the complement of every maximal outerplanar graph with seven points is not outer planar.

This holds because there are exactly four maximal outer planar graphs with P = 7. (See Figure above) and the complement of each is readily seen to be non outer planar.

The minimality follows from the fact that the (maximal) outer planar graph of Figure below, with six points has an outer planar complement.

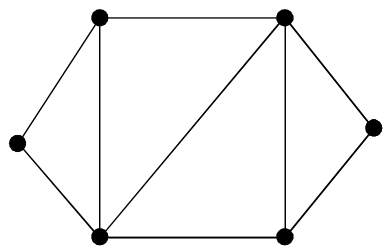


Fig. 2.42.

Lemma 1.

There is a cycle in F containing u_0 and v_0.

Proof. Assume that there is no cycle in F containing u_0 and v_0.

Then u_0 and v_0 lie in different blocks of F.

Hence, there exists a cut point W of F lying on every $u_0 - v_0$ path.

We form the graph F_0 by adding to F the lines wu_0 and wv_0 if they are not already present in F.

In the graph F_0, u_0 and v_0 still lie in different blocks, say B_1 and B_2, which necessarily have the point W in common. Certainly, each of B_1 and B_2 has fewer lives than G, so either B_1 is planar or it contains a subgraph homeomorphic to K_5 or $K_{3,3}$.

If, however, the insertion of wu_0 produces a subgraph H of B_1 homeomorphic to K_5 or $K_{3,3}$, then the subgraph of G obtained by replacing wu_0 by a path from u_0 to W which begins with x_0 is necessarily homeomorphic to H and so to K_5 or $K_{3,3}$, but this is a contradiction.

Hence, B_1 and similarly B_2 is planar. Both B_1 and B_2 can be drawn in the plane so that the lines wu_0 and wv_0 bound the exterior region.

Hence it is possible to embed the graph F_0 in the plane with both wu_0 and wv_0 on the exterior region.

Inserting x_0 cannot then destroy the planarity of F_0. Since G is a subgraph of $F_0 + x_0$, G is planar, this contradiction shows that there is a cycle in F containing u_0 and v_0.

Let F be embedded in the plane in such a way that a cycle Z containing u_0 and v_0 has a maximum number of regions interior to it.

Orient the edges of Z in a cyclic fashion, and let Z[u, v] denote the oriented path from u to v along Z.

If v does not immediately follow u to z, we also write Z(u, v) to indicate the subgraph of Z[u, v] obtained by removing u and v.

By the exterior of cycle Z, we mean the subgraph of F induced by the vertices lying outside Z, and the components of this subgraph are called the exterior components of Z.

By an outer piece of Z, we mean a connected subgraph of F induced by all edges incident with atleast one vertex in some exterior component or by an edge (if any) exterior to Z meeting two vertices of Z. In a like manner, we define the interior of cycle Z, interior component, and inner piece.

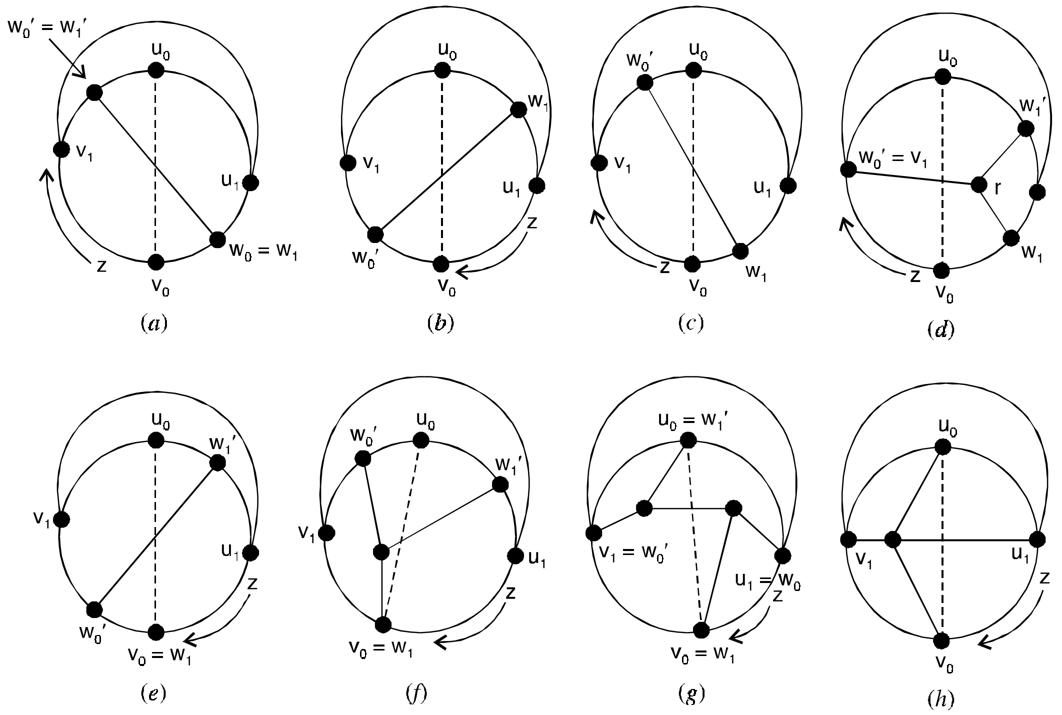


Fig. 2.43. Separating cycle Z illustrating lemma.

An outer or inner piece is called $u - v$ separating if it meets both Z(u, v) and Z(v, u).

Clearly, an outer or inner piece cannot be $u - v$ separating if u and v are adjacent on Z.

Since F is connected, each outer piece must meet Z, and because F has no cut vertices, each outer piece must have atleast two vertices in common with Z.

No outer piece can meet Z(u_0, v_0) or Z(v_0, u_0) in more than one vertex, for otherwise there would exist a cycle containing u_0 and v_0 with more interior regions than Z.

For the same region, no outer piece can meet u_0 or v_0.

Hence every outer piece meets Z in exactly two vertices and is $u_0 - v_0$ separating.

Further more, since x_0 cannot be added to F in planar fashion, there is at least one $u_0 - v_0$ separating inner piece.

Lemma 2.

There exists a $u_0 - v_0$ separating outer piece meeting Z(u_0, v_0), say at u_1, and Z(v_0, u_0), say at v_1, such that there is an inner piece which is both $u_0 - v_0$ separating and $u_1 - v_1$ separating.

Proof. Suppose, to the contrary, that the lemma does not hold. It will be helpful in understanding this proof to refer to Figure (2.43).

We order the $u_0 - v_0$ separating inner pieces for the purpose of relocating them in the plane. Consider any $u_0 - v_0$ separating inner piece I_1 which is nearest to u_0 in the sense of encountering points of this inner piece on moving along Z from u_0. Continuing out from u_0, we can index the $u_0 - v_0$ separating inner pieces I_2, I_3 and so on.

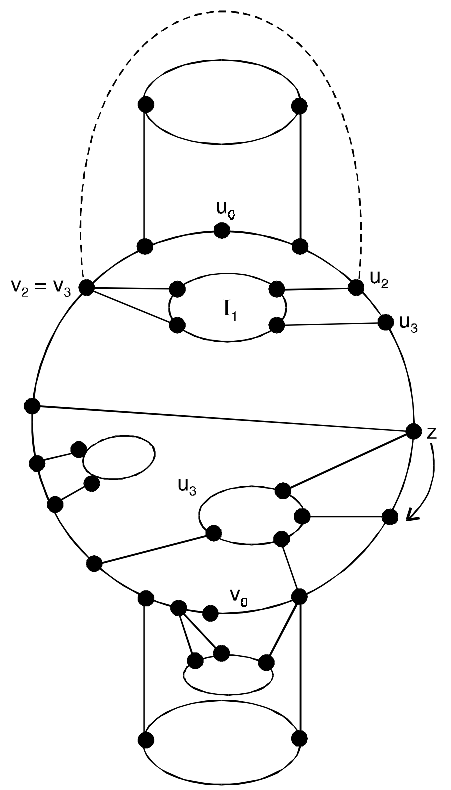


Fig. 2.44. The possibilities for non planar graphs.

Let u_2 and u_3 be the first and last points of I_1 meeting $Z(u_0, v_0)$ and v_2 and v_3 be the first and last vertices of I_1 meeting $Z(v_0, u_0)$.

Every outer piece necessarily has both its common vertices with Z on either $Z[v_3, u_2]$ or $Z[u_2, v_2]$, for otherwise, there would exist an outer piece meeting $Z(u_0, v_0)$ at u_1 and $Z(v_0, u_0)$ at v_1 and an inner piece which is both $u_0 - v_0$ separating and $u_1 - v_1$ separating, contrary to the supposition that the lemma is false.

Therefore, a curve C joining v_3 and u_2 can be drawn in the exterior region so that it meets no edge of F (see Figure (2.43).

Thus, I_1 can be transferred outside of C in a planar manner.

Similarly, the remaining $u_0 - v_0$ separating inner pieces can be transferred outside of Z, in order, so that the resulting graph is plane.

However, the edge x_0 can then be added without destroying the planarity of F, but this is a contradiction, completing the lemma.

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.

Theorem 2.17. *A graph is planar if and only if it does not have a subgraph contractible to K_5 or $K_{3,3}$.*

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 $e \geq 7$.

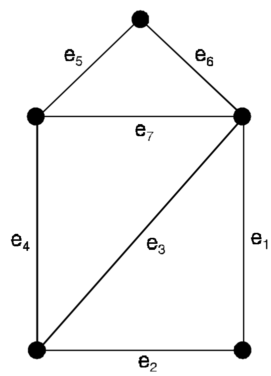


Fig. 2.45.

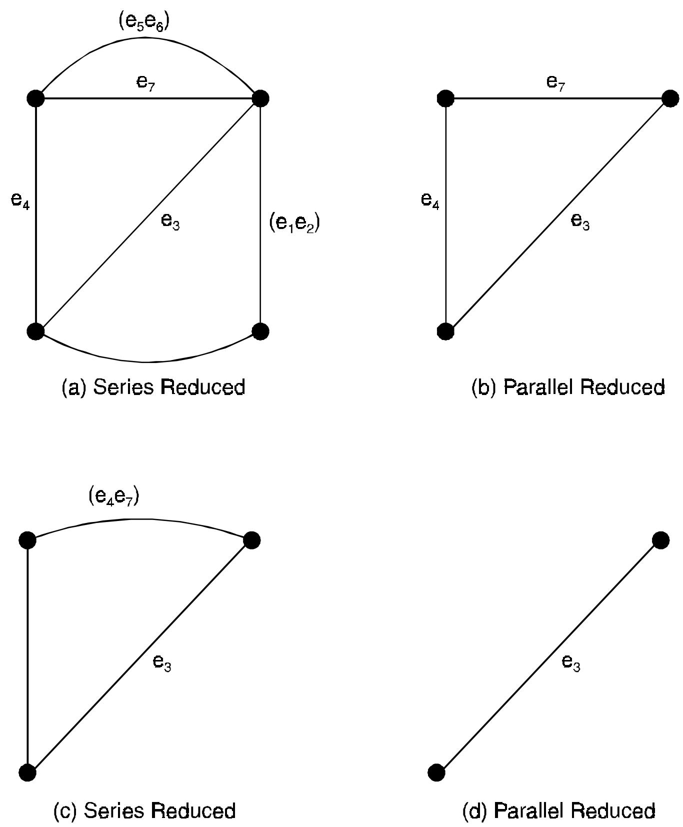


Fig. 2.46. Series-parallel reduction of the graph in Figure 2.45

Problem 2.35. *Check the planarity of the following graph by the method of elementary deduction.*

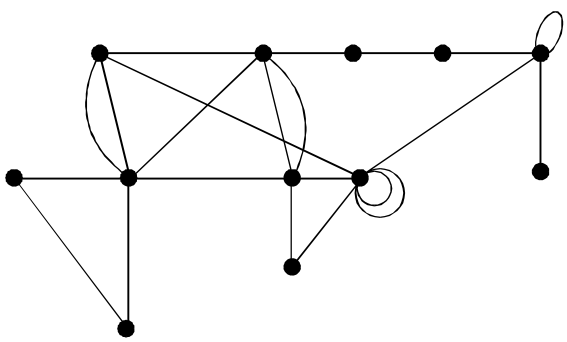


Fig. 2.47.

Solution. Step 1 : Does not apply, because the graph is connected.

Step 2 : Separating blocks of G

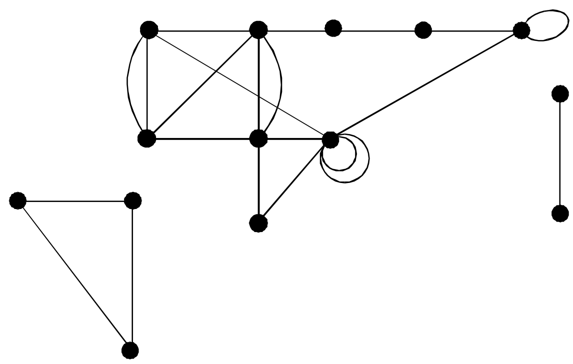


Fig. 2.48.

Step 3 : Removing self-loops and parallel edges

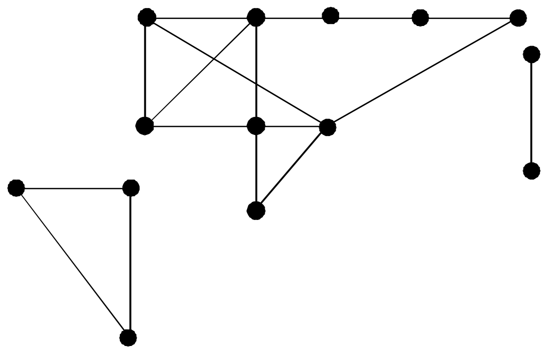


Fig. 2.49.

Step 4 : Merging the series edges.

The final graph contains three components. Largest component contains 7 vertices. Remaining two is triangle and an edge hence they are planar. The largest component contains no subgraph isomorphic to K_5 or $K_{3,3}$ and hence it is planar.

Thus the given graph is planar.

Problem 2.36. *Check the planarity of the following graph by the method of elementary reduction.*

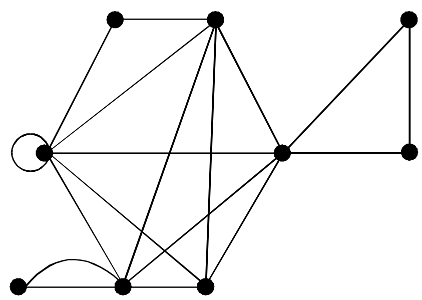


Fig. 2.50.

Solution. The elementary reduction of the given graph G consists of the following stages :

Step 1 : Splitting G into blocks. This splitting is shown below :

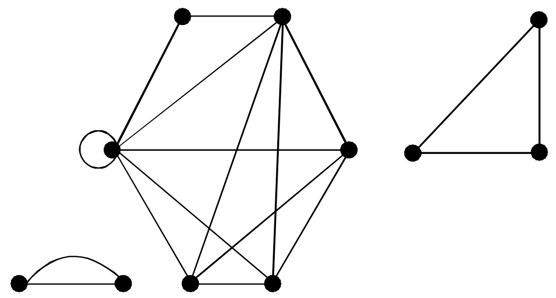


Fig. 2.51.

Step 2 : Removing self-loops and eliminating multiple edges. The resulting graph is as shown below :

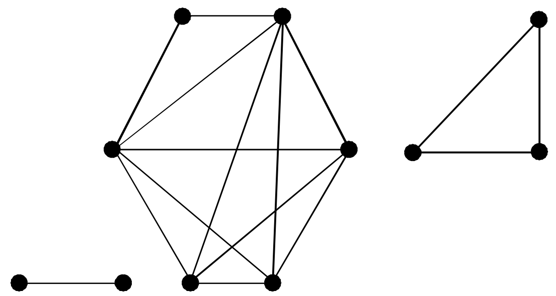


Fig. 2.52.

Step 3 : Merging the edges incident on vertices of degree 2. The resulting graph is as shown below :

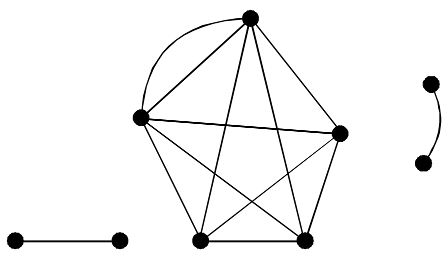


Fig. 2.53.

Step 4 : Eliminating parallel edges. The resulting graph is shown below :

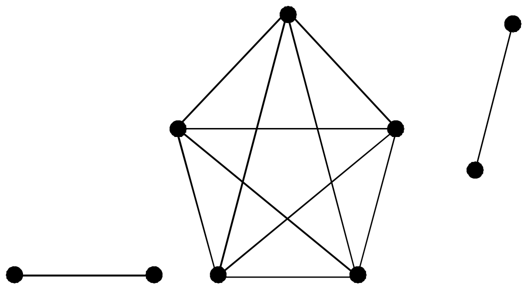


Fig. 2.54.

The reduction is now complete. The final reduced graph (shown in Figure above) has three blocks, of which the first and the third are obviously planar. The second one is evidently the complete graph K_5, which is non planar.

Thus, the given graph contains K_5 as a subgraph and is therefore non planar.

Problem 2.37. *Carryout the elementary reduction process for the following graph :*

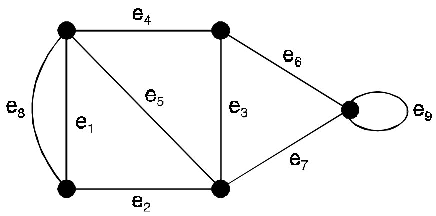


Fig. 2.55.

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 :

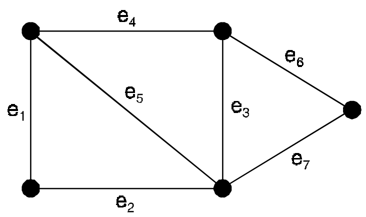


Fig. 2.56.

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 :

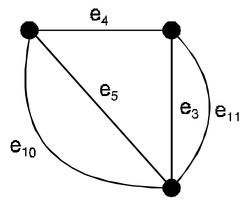


Fig. 2.57.

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 :

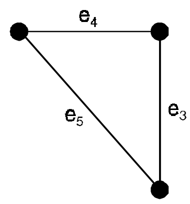


Fig. 2.58.

As per step 4, we merge the edges e_3 and e_4 into an edge e_{12} (say) to get the following graph.

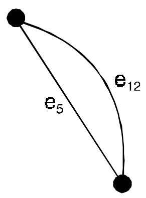
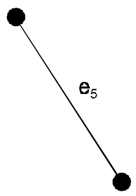


Fig. 2.59.

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 DUAL OF A PLANAR GRAPH

Consider the plane representation of a graph in Figure (2.60)(*a*) with six regions of faces F_1, F_2, F_3, F_4, F_5 and F_6.

Let us place six points P_1, P_2, ... P_6, one in each of the regions, as shown in Figure (2.60)(*b*).

Next let us join these six points according to the following procedure :

(*i*) If two regions F_i and F_j are adjacent (*i.e.,* have a common edge), draw a line joining points P_i and P_j that intersects the common edge between F_i and F_j exactly once.

(*ii*) If there is more than one edge common between F_i and F_j, draw one line between points P_i and P_j for each of the common edges.

(*iii*) For an edge *e* lying entirely in one region, say F_k, draw a self-loop at point P_k intersecting *e* exactly once.

By this procedure we obtained a new graph G* (in broken lines in Figure (2.60)(*c*) consisting of six vertices, P_1, P_2, ... P_6 and of edges joining these vertices. Such a graph G* is called **dual** (a geometrical dual) of G

Clearly, there is a one-to-one correspondence between the edges of graph G and its dual G*— one edge of G* intersecting one edge of G. Some simple observations that can be made about the relationship between a planar graph G and its dual G* are :

(*i*) An edge forming a self-loop in G yields a pendant edge in G*.

(*ii*) A pendant edge in G yields a self-loop in G*.

(*iii*) Edges that are in series in G produce parallel edges in G*.

(*iv*) Parallel edges in G produce edges in series in G*.

(*v*) Remarks (*i*)-(*iv*) are the result of the general observation that the number of edges constituting the boundary of a region F_i in G is equal to the degree of the corresponding vertex P_i in G*.

(*vi*) Graph G* is also embedded in the plane and is therefore planar.

(*vii*) Considering the process of drawing a dual G* from G, it is evident that G is a dual of G* (see Fig. (2.60) (*c*)). Therefore, instead of calling G* a dual of G, we usually say that G and G* are dual graphs.

(*viii*) If n, e, f, r and μ denote as usual the numbers of vertices, edges, regions, rank, and nullity of a connected planar graph G, and if n^*, e^*, f^*, r^* and μ^* are the corresponding numbers in dual graph G*, then

$$n^* = f,\ e^* = e,\ f^* = n.$$

Using the above relationship, one can immediately get $r^* = \mu,\ \mu^* = r$.

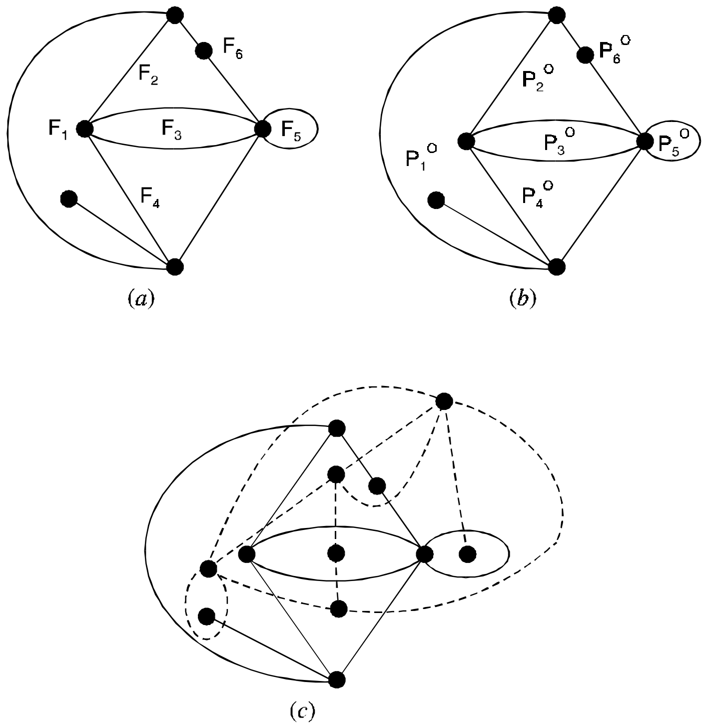


Fig. 2.60. Construction of a dual graph.

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*).

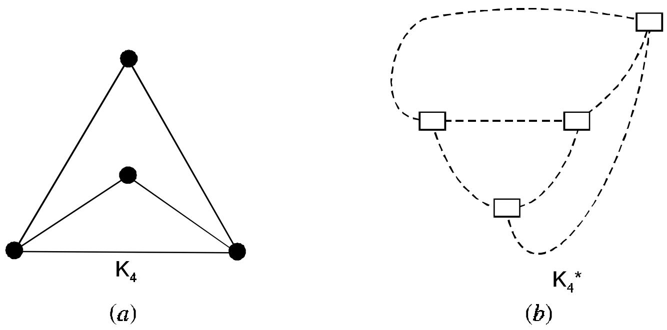


Fig. 2.61.

We observe that K_4* has four vertices and six edges. Also, every two vertices of K_4* are joined by an edge. This means that K_4* also represents the complete graph of four vertices. As such, K_4 and K_4* are isomorphic. In other words, K_4 is a self-dual graph.

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*.

2.14.6. Abstract dual

Given two graphs G_1 and G_2, we say that G_1 and G_2 are abstract duals of each other if there is a one-to-one correspondence between the edges in G_1 and the edges in G_2, with the property that a set of edges in G_1 forms a circuit in G_1 if and only if the corresponding set of edges in G_2 forms a cut-set in G_2.

Consider the graphs G_1 and G_2 shown in Figure (2.62).

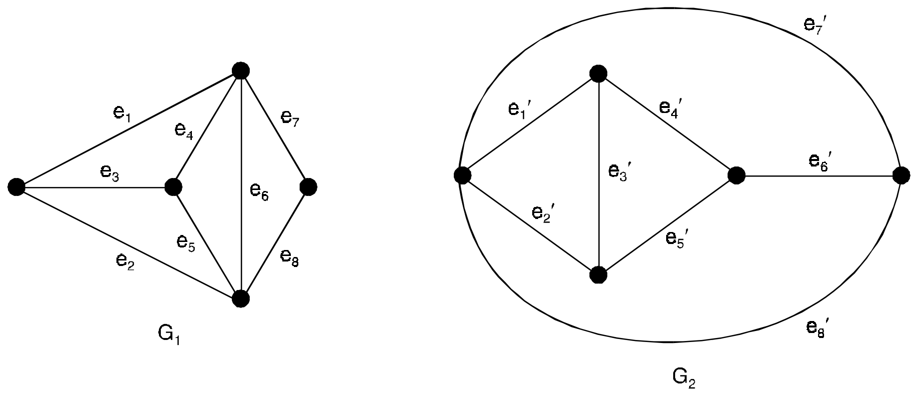


Fig. 2.62.

We observe that there is a one-to-one correspondence between the edges in G_1 and the edges in G_2 with the edge e_i in G_1 corresponding to the edge e_i' in G_2, $i = 1, 2, ... 8$.

Further, note that a set of edges in G_1 which forms a circuit in G_1 corresponds to a set of edges in G_2 which forms a cut sets in G_2.

For example, $\{e_6, e_7, e_8\}$ is a circuit in G_1 and $\{e_6', e_7', e_8')$ is a cut-set in G_2.

Accordingly, G_1 and G_2 are abstract duals of each other.

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

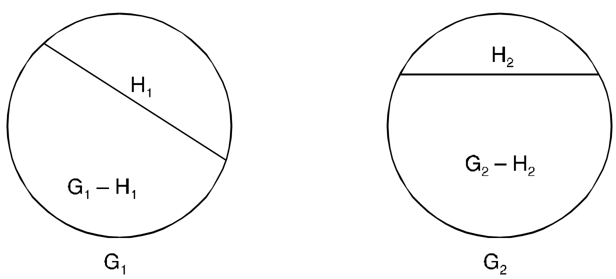


Fig. 2.63.

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*).

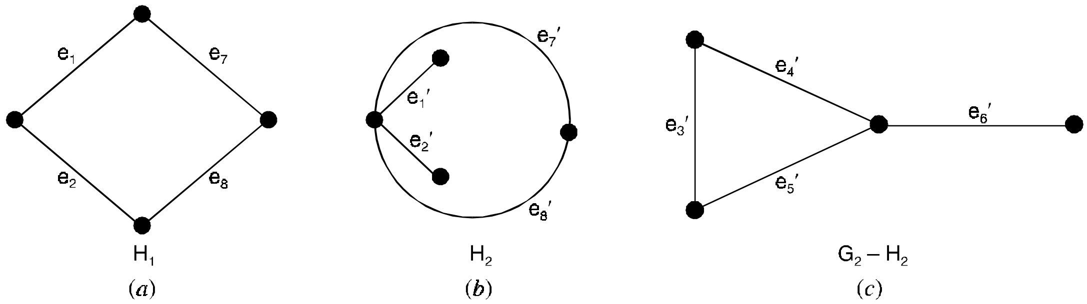


Fig. 2.64.

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 2.18. *If G is a plane connected graph, then G** is isomorphic to G.*

Proof. The result follows immediately, since the construction that gives rise to G* from G can be reversed to give G from G*,

For example, in Figure (2.65), the graph G is the dual of the graph G*

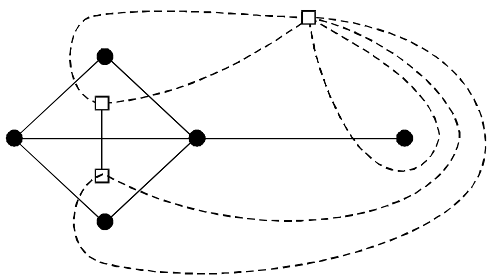


Fig. 2.65.

We need to check only that a face of G* cannot contain more than one vertex of G (it certainly contains at least one) and this follows immediately from the relations $n^{**} = f^{*} = n$, where n^{**} is the number of vertices of G**.

Theorem 2.19. *Let G be a planar graph and let G* be a geometric dual of G. Then a set of edges in G forms a cycle in G if and only if the corresponding set of edges of G* forms a cutset in G*.*

Proof. We can assume that G is a connected plane graph. If C is a cycle in G, then C encloses one or more finite faces C, and thus contains in its interior a non-empty set S of vertices of G*.

It follows immediately that choose edges of G* that cross the edges of C form a cutset of G* whose removal disconnects G* into two subgraphs, one with vertex set S and the other containing those vertices that do not lie in S (see Figure 2.66).

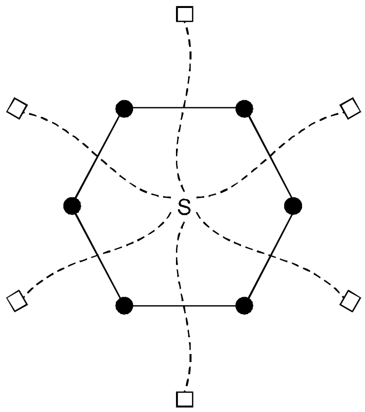


Fig. 2.66.

Corollary : A set of edges of G forms a cutset in G if and only if the corresponding set of edges of G* forms a cycle in G*.

Theorem 2.20. *If G* is an abstract dual of G, then G is an abstract dual of G*.*

Proof. Let C be a cutset of G and let C* denote the corresponding set of edges of G*.

We show that C* is a cycle of C*.

C has an even number of edges in common with any cycle of G, and so C* has an even number of edges in common with any cut set of G*.

C* is either a cycle in G* or an edge-disjoint union of at least two cycles.

But the second possibility cannot occur, since we can show similarly that cycles in C* correspond to edge-disjoint unions of cut sets in G, and so C would be an edge-disjoint union of at least two cutsets, rather than a single cutset.

Theorem 2.21. *A graph is planar if and only if it has an abstract dual.*

Proof. It is sufficient to prove that if G is a graph with an abstract dual G*, then G is planar. The proof is in four steps.

(*i*) We note first that if an edge e is removed from G, then the abstract dual of the remaining graph may be obtained from G* by contracting the corresponding edge e^*.

On the repeating this procedure, we deduce that, if G has an abstract dual, then so does any subgraph of G.

(*ii*) We next observe that if G has an abstract dual, and G′ is homeomorphic to G, then G′ also has an abstract dual.

This follows from the fact that the insertion or removal in G of a vertex of degree 2 results in the addition or deletion of a multiple edge in G*.

(*iii*) The third step is to show that neither K_5 nor $K_{3,3}$ has an abstract dual.

If G* is a dual of $K_{3,3}$ then since $K_{3,3}$ contains only cycles of length 4 or 6 and no cutsets with two edges, G* contains no multiple edges and each vertex of G* has degree at least 4.

Hence G* be have at least five vertices, and thus atleast $\frac{(5\times 4)}{2} = 10$ edges, which is a contradiction.

The argument for K_5 is similar and is omitted.

(*iv*) Suppose, now, that G is a non-planar graph with an abstract dual G*.

Then, by Kuratowski's theorem, G has a subgraph H homeomorphic to K_5 to $K_{3,3}$.

It follows from (*i*) and (*ii*) that H, and hence also K_5 or $K_{3,3}$, must have an abstract dual, contradicting (*iii*).

Theorem 2.22. *Let G be a connected planar graph with n vertices, m edges and r regions, and let its geometric dual G* have n* vertices, m* edges and r* regions. Then n* = r, m* = m, r* = n.*

Further, if ρ *and* ρ^* *are the ranks and* μ *and* μ^* *are the nullities of G and G* respectively, then* $\rho^* = \mu$ *and* $\mu^* = \rho$.

Proof. Every region of G yields exactly one vertex of G* and G* has no other vertex.

Hence the number of regions in G is precisely equal to the number of vertices of G*,

i.e., $r = n^*$. ...(1)

Corresponding to every edge e of G, there is exactly one edge e^* of G* that crosses e exactly once, and G* has no other edge.

Thus G and G* have the same number of edges,

i.e., $m = m^*$ ...(2)

Now, the Euler's formula applied to G* and G yields

$$\begin{aligned} r^* &= m^* - n^* + 2 \\ &= m - r + 2 \\ &= n \end{aligned}$$

Since G and G* are connected, we have

$$\rho = n - 1, \quad \mu = m - n + 1$$
$$\rho^* = n^* - 1, \quad \mu^* = m^* - n^* + 1$$

These together with the results (1) and (2) and the Euler's formula yield

$$\begin{aligned} \rho^* &= n^* - 1 = r - 1 = (m - n + 2) - 1 \\ &= m - n + 1 = \mu \end{aligned}$$
$$\begin{aligned} \mu^* &= m^* - n^* + 1 = m - r + 1 \\ &= m - (m - n + 2) + 1 = n - 1 = \rho. \end{aligned}$$

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.

Problem 2.38. *If G is a 3-connected planar graph, prove that its geometric dual is a simple graph.*

Solution. If G is 3-connected, then G has no vertices of degree 1 or 2.

Therefore, G* has no self-loops or multiple edges. That is, G* is simple.

Problem 2.39. *Show that a connected planar self-dual graph G with n vertices should have 2n – 2 edges.*

Solution. Since the graph G is self-dual, we have $n = n^*$. But $n^* = r$,

Therefore, in G, $n = r$,

The Euler's formula now gives $n = m - n + 2$

or $$m = 2n - 2.$$

Problem 2.40. *Show that a set of edges in a connected planar graph G forms a spanning tree of G if and only if the set of duals of the remaining edges forms a spanning tree of a geometric dual of G.*

Solution. Consider a connected planar graph G with n vertices and m edges.

Let T be a spanning tree of G. This is a set of $n - 1$ edges. The remaining edges are $m - (n - 1)$ in number.

The duals of these edges are also $m - (n - 1)$ in number.

The set T* of these duals belong to G*.

Since G* has $m - n + 2$ vertices, the set T* which consists of $m - n + 1$ vertices is a spanning tree of G*.

This proves the first part of the required result.

By reversing the roles of G and G* in the above argument, we get the second proof.

Problem 2.41. *Show that there is no planar graph with five regions such that there is an edge between every pair of regions.*

Solution. Suppose there is a planar graph G having the desired property.

Then, the geometric dual G* of G will have five vertices such that there is an edge between every pair of vertices.

This means that G* is the graph K_5.

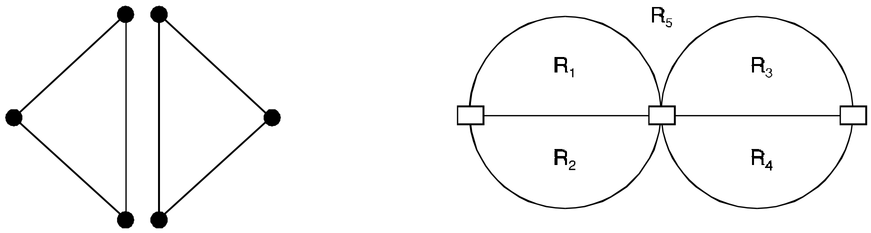
Therefore, G* is non planar.

This is a contradiction because G* has to be planar. (like G).

Hence, a planar graph of the desired type does not exist.

Problem 2.42. *Disprove that the geometric dual of the geometric dual of a planar graph G is the same as the abstract dual of the abstract dual of G.*

Solution. Consider the disconnected graph G with two components, each of which is a triangle as shown in Figure (2.67)(*a*).



(*a*) (*b*)

Fig. 2.67.

The geometric dual G* is shown in Figure (2.67)(*b*), we observe that G* has five regions.

Therefore, the geometric dual G** of G* has five vertices.

On the other hand, if G′ is the abstract dual of G, then G is the abstract dual of G′.

Hence, G is the abstract dual of the abstract dual of G. *i.e.*, G = G″.

Since G has six vertices, it follows that G″ cannot be the same as G** (which has five vertices).

The above counter example disproves that the geometric dual of the geometric dual is the same as the abstract dual of the abstract dual.

Problem 2.43. *Let G be a connected planar graph. Prove that G is bipartite if and only if its dual is on Euler graph.*

Solution. If G is bipartite, then each circuit of G has even length.

Therefore, each cutset of its dual G′ has an even number of edges.

In particular, each vertex of G′ has even degree.

Therefore G′ is an Euler graph.

Theorem 2.24. *Let G be a plane connected graph. Then G is isomorphic to its double dual G**.*

Proof. Let *f* be any face of the dual G* contains atleast one vertex of G, namely its corresponding vertex *v*.

In fact this is the only vertex of G that *f* contains since by theorem.

i.e., a connected graph G with *n*-vertices, *e*-edges, *f*-faces and n^*, e^*, f^* denotes the vertices, edges and faces of G* then $n^* = f$, $e^* = e$, $f^* = n$, the number of faces of G* is the same as the number of vertices of G.

Hence in the construction of double dual G**, we may choose the vertex *v* to be the vertex in G** corresponding to face *f* of G*.

This choice gives our required result.

Theorem 2.25. *Let G be a connected plane graph with n-vertices e-edges and f-faces. Let n^*, e^* and f^* denote the number of vertices, edges and faces respectively of G^*, then $n^* = f$, $e^* = e$ and $f^* = n$.*

Proof. The first two relations are direct consequence of the definition of G, the third relation follows immediately on substituting these two relations into Euler's theorem applied to both G and G*.

If G is a plane graph then G* is also a plane graph. We may also construct the dual of G*, called the double dual of G and denoted by G**.

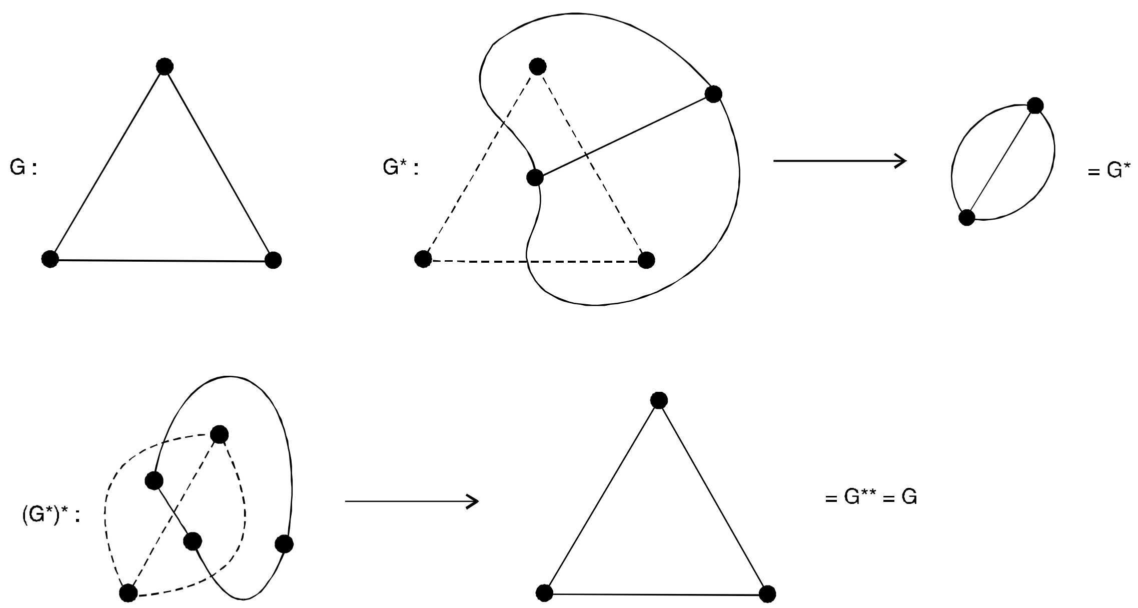


Fig. 2.68.

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.

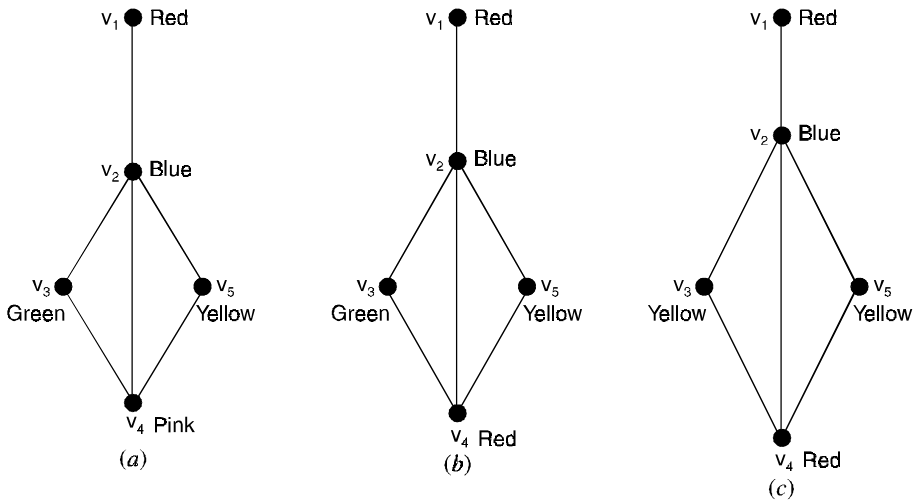


Fig. 2.69. Proper colours of a graph.

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.

2.15.4. K-Critical graph

If the chromatic number denoted by (G) = K, and (G – v) is less than equal to K – 1 for each vertex v of G, then

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).*

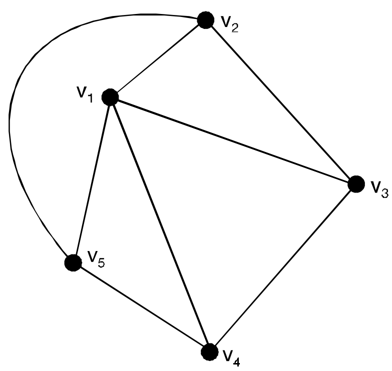


Fig. 2.70. A 3-chromatic graph.

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.

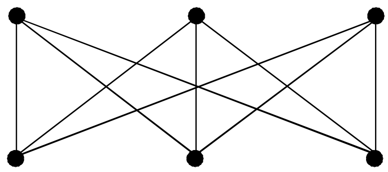
Therefore, C_4 = 24 • 2 = 48.

Substituting these coefficients in $P_5(\lambda)$, we get, for the graph in Figure (2.70).

$$P_5(\lambda) = \lambda(\lambda - 1)(\lambda - 2) + 2\lambda(\lambda - 1)(\lambda - 2)(\lambda - 3) + \lambda(\lambda - 1)(\lambda - 2)(\lambda - 3)(\lambda - 4)$$
$$= \lambda(\lambda - 1)(\lambda - 2)(\lambda^2 - 5\lambda + 7)$$

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.

Problem 2.46. *Find the chromatic polynomial and hence the chromatic number for the graph shown below.*

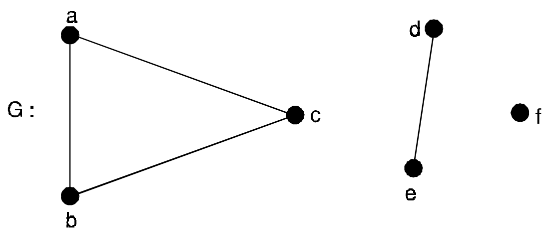


Fig. 2.71.

Solution. Since G is made up of components of G_1, G_2 and G_3 where $G_1 = K_3$, G_2 is a linear graph and G_3 is an isolated vertex.

Now G_1 can be coloured in $\lambda(\lambda - 1)(\lambda - 2)$ ways, G_2 can be coloured in $\lambda(\lambda - 1)$ ways and G_3 is λ ways.

Therefore, by the rule of product G can be coloured be

$$\lambda(\lambda - 1)(\lambda - 2)\lambda(\lambda - 1)\lambda = \lambda^3(\lambda - 1)^2(\lambda - 2).$$

2.16.1. Decomposition theorem (2.26)

If $G = (V, E)$ is a connected graph and $e = \{a, b\} \in E$, then $P(G_e, \lambda) = P(G, \lambda) + P(G_e', \lambda)$.

Where G_e denotes the subgraph of G obtained by deleting *e* from G without removing vertices *a* and *b*.

i.e., $G_e = G - e$ and G_e' is a second subgraph of G obtained from G_e' by colouring the vertices *a* and *b*.

Proof. Let $e = \{a, b\}$. The number of ways to properly color the vertices in $G_e = G - e$ with (atmost) λ colours in $P(G_e, \lambda)$.

Those colourings where end points *a* and *b* of *e* have different colours are proper colourings of G.

The colourings of G_e that are not proper colourigns of G occur when *a* and *b* have the same color.

But each of these colourings corresponds with a proper colouring for G_e'.

This partition of the $P(G_e, \lambda)$ proper colourings of G_e into two disjoint subsets results in the equation

$$P(G_e, \lambda) = P(G, \lambda) + P(G_e', \lambda)$$

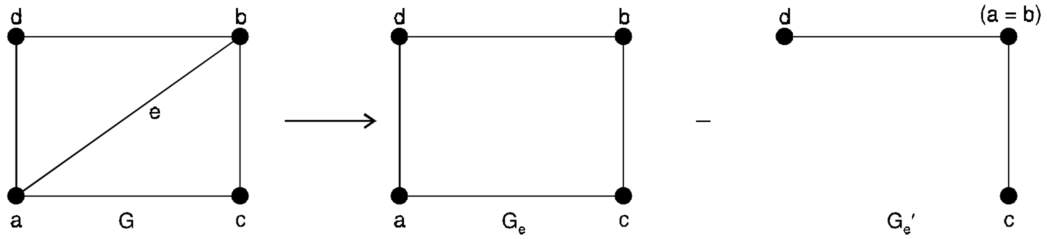
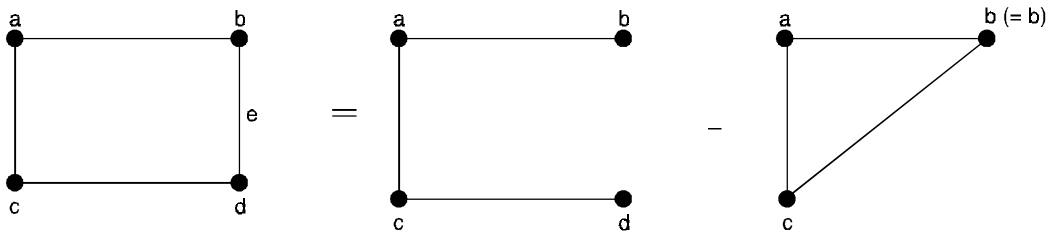


Fig. 2.72.

Problem 2.47. *Using decomposition theorem find the chromatic polynomial and hence the chromatic number for the graph given below in Figure (2.73).*



(*a*) (*b*) (*c*)

Fig. 2.73.

Solution. Deleting the edge *e* from G, we get G_2 as shown in Figure (*b*). Then the chromatic polynomial of G_e is

$$P(G_e, \lambda) = \lambda(\lambda - 1)(\lambda - 2)$$

By colouring the endpoints of *e*, *i.e.*, *a* and *b*, we get G_e' as shown in Figure (*c*). Then the chromatic polynomial of G_e is

$$P(G_e', \lambda) = \lambda(\lambda - 1)^3.$$

Hence, by decomposition theorem, the chromatic polynomial of G is

$$\begin{aligned} P(G, \lambda) &= \lambda(\lambda - 1)^3 - \lambda(\lambda - 1)(\lambda - 2) \\ &= \lambda(\lambda - 1)\ [(\lambda - 1)^2(\lambda - 2)] \\ &= \lambda(\lambda - 1) - (\lambda^3 - 3\lambda + 3)\ \lambda^4 \\ &= 4\lambda^3 + 6\lambda - 3\lambda. \end{aligned}$$

Theorem 2.27. *For each graph G, the constant term in $P(G, \lambda)$ is 0.*

Proof. For each graph G, $\lambda(G) > 0$ because $V \neq \phi$.

If $P(G, \lambda)$ has constant term a, then $P(G, 0) = a \neq 0$.

This implies that there are a ways to colour G properly with 0 colours, a contradiction.

Theorem 2.28. *Let $G = (V, E)$ with $| E | > 0$. Then the sum of the coefficients in $P(G, \lambda)$ is 0.*

Proof. Since $| E | \geq 1$, we have $\lambda(G) \geq 2$, so we cannot properly colour G with only one colour.

Consequently, $P(G, 1) = 0$ = the sum of the coefficients in $P(G, \lambda)$.

Problem 2.48. *Explain why each of the following polynomials cannot be a chromatic polynomial*

(*i*) $\lambda^3 + 5\lambda^2 - 3\lambda + 5 = 0$

(*ii*) $\lambda^4 + 3\lambda^3 - 3\lambda^2 = 0.$

Solution. (*i*) The polynomial cannot be a chromatic polynomial since the constant term is 5, not 0.

(*ii*) The polynomial cannot be a chromatic polynomial since the sum of the coefficient is 1, not 0.

Theorem 2.29. *(Vizing) If G is a simple graph with maximum vertex degree Δ then $\Delta \leq \chi'(G) \leq \Delta + 1$.*

Theorem 2.30. *Let $\Delta(G)$ be the maximum of the degrees of the vertices of a graph G. Then $\chi(G) \leq 1 + \Delta(G)$.*

Proof. The proof is by induction on V, the number of vertices of the graph.

When $V = 1$, $\Delta(G) = 0$ and $\chi(G) = 1$, so the result clearly holds.

Now let K be an integer $K \geq 1$, and assume that the result holds for all graphs with $V = K$ vertices.

Suppose G is a graph with $K + 1$ vertices.

Let v be any vertex of G and let $G_0 = \dfrac{G}{\{v\}}$ be the subgraph with v (and all edges incident with it) deleted.

Note that $\Delta(G_0) \leq \Delta(G)$. Now G_0 can be be coloured with $\chi(G_0)$ colours.

Since G_0 has K vertices, we can use the induction hypothesis to conclude that $\chi(G_0) \leq 1 + \Delta(G_0)$.

Thus, $\chi(G_0) \leq 1 + \Delta(G)$, so go can be coloured with atmost $1 + \Delta(G)$ colours.

Since there are atmost $\Delta(G)$ vertices adjacent to v, one of the variable $1 + \Delta(G)$ colours remains for v.

Thus, G can be coloured with atmost $1 + \Delta(G)$ colours.

Theorem 2.31. (Kempe, Heawood). *If G is a planar graph, then $\chi(G) \leq 5$.*

Proof. We must prove that any planar graph with V vertices has a 5-colouring.

Again we use induction on V and note that if V = 1, the result is clear.

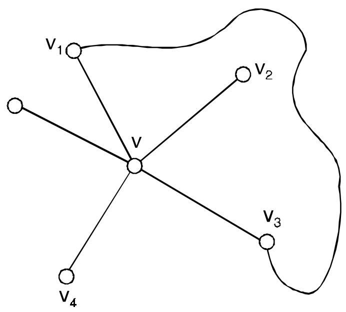


Fig. 2.74.

Let K ≥ 1 be an integer and suppose that any planar graph with K vertices has a 5-colouring.

Let G be a planar graph with K + 1 vertices and assume that G has been drawn as a plane graph with straight edges. We describe how to obtain a 5-colouring of G.

First, G contains a vertex v of degree atmost 5.

Let $G_0 = \dfrac{G}{\{v\}}$ be the subgraph obtained by deleting v (and all edges with which it is incident).

By the induction hypothesis, G_0 has a 5-colouring.

For convenience, label the five colours 1, 2, 3, 4 and 5.

If one of these colours was not used to colour the vertices adjacent to v, then it can be used for v and G has been 5-coloured.

Thus, we assume that v has degree 5 and that each of the colour 1 through 5 appears on the vertices adjacent to v.

In clockwise order, label these vertices v_1, v_2, ... v_5 and assume that v_i is coloured with colour i (see Figure 2.74).

We show how to recolour certain vertices of G_0 so that a colour becomes available for v.

There are two possibilities :

Case 1 : There is no path in G_0 from v_1 to v_3 through vertices all of which are coloured 1 or 3.

In this situation, let H be the subgraph of G consisting of the vertices and edges of all paths through vertices coloured 1 or 3 which start at v_1.

By assumption, v_3 is not in H. Also, any vertex which is not in H but which is adjacent to a vertex of H is coloured neither 1 nor 3.

Therefore, interchanging colours 1 and 3 throughout H produces another 5-colouring of G_0.

In this new 5-colouring both v_1 and v_3 acquire colour 3, so we are now free to give color 1 to v, thus obtaining a 5-colouring of G.

Case 2 : There is a path P in G_0 from v_1 to v_3 through vertices all of which are coloured 1 or 3.

In this case, the path P, followed by v and v_1, gives a circuit in G which does not enclose both v_2 and v_4. Thus, any path from v_2 to v_4 must cross P and, since G is a plane graph, such a crossing can occur only at a vertex of P.

It follows that there is no path in G_0 from v_2 to v_4 which uses just colours 2 and 4.

Now we are in the situation described in case (1), where we have already shown that a 5-colouring for G exists.

Problem 2.49. *$\chi(K_n) = n$, $\chi(K_{m,\,n}) = 2$, why ?*

Solution. It takes *n* colours to colour K_n because any two vertices of K_n are adjacent. $\chi(K_n) = n$.

On the otherhand, $\chi(K_{m,\,n}) = 2$, colouring the vertices of each bipartition set the same colour produces a 2-colouring of $K_{m,\,n}$.

Problem 2.50. *What is the chromatic number of the graph in Figure (2.75).*

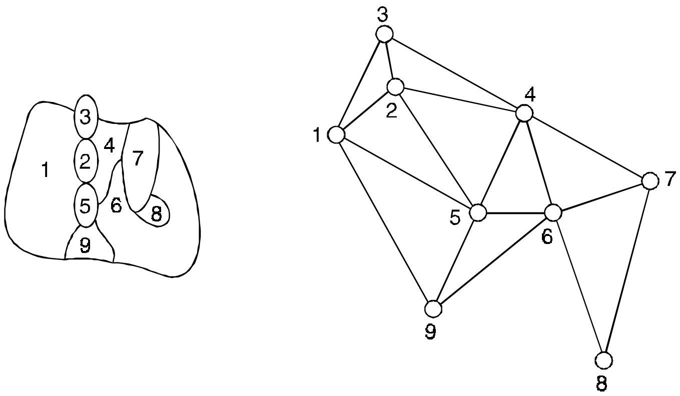


Fig. 2.75. A map and an associated planar graph.

Solution. A way to 4-colour the associated graph, was given in the text. From this, we deduce that $\chi(G) \leq 4$.

To see that $\chi(G) = 4$, we investigate the consequences of using fewer than four colours.

Vertices 1, 2, 3 from a triangle, so three different colours are needed for these.

Suppose we use red, blue and green, respectively, as before.

To avoid a fourth colour, vertex 4 has to be coloured red and vertex 5 green.

Thus, vertex 6 has to be blue.

Since vertex 9 is adjacent to vertices 1, 5 and 6 of colours red, green and blue, respectively. Vertex 9 requires a fourth colour.

Problem 2.51. *Show that $\chi(G) = 4$ for the graph of G of Figure (2.76).*

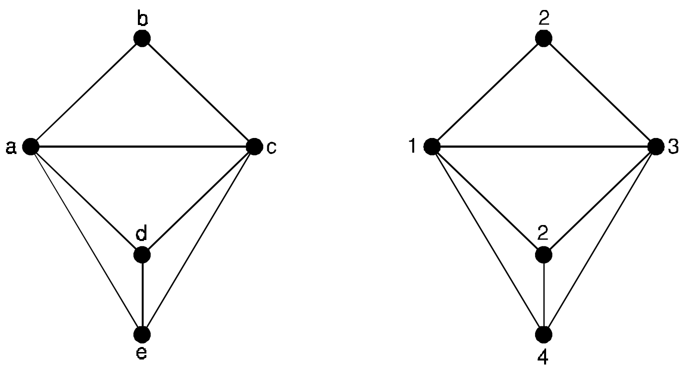


Fig. 2.76.

Solution. Clearly the triangle *abc* requires three colours, assign the colours 1, 2 and 3 to *a*, *b* and *c* respectively.

Then since *d* is adjacent to *a* and *c*, *d* must be assigned a colour different from the colours for *a* and *c*, colour *d* is colour 2.

But then *e* must be assigned a colour different from 2 since *e* is adjacent to *d*.

Likewise *e* must be assigned a colour different from 1 or 3 because *e* is adjacent to *a* and to *c*.

Hence a fourth colour must be assigned to *e*.

Thus, the 4-colouring exhibited incidates $\chi(G) \leq 4$.

But, at the same time, we have argued that $\chi(G)$ cannot be less than 4.

Hence $\chi(G) = 4$.

Theorem 2.32. *The minimum number of hours for the schedule of committee meetings in our scheduling problem is* $\chi(G_0)$.

Proof. Suppose $\chi(G_0) = K$ and suppose that the colours used in colouring G_0 are 1, 2, ... K.

First we assert that all committees can be scheduled in K one-hour time periods.

In order to see this, consider all those vertices coloured 1, say, and the committees corresponding to these vertices.

Since no two vertices coloured 1 are adjacent, no two such committees contain the same member.

Hence, all these committees can be scheduled to meet at the same time.

Thus, all committees corresponding to same-coloured vertices can meet at the same time.

Therefore, all committees can be scheduled to meet during K time periods.

Next, we show that all committees cannot be scheduled in less than K hours. We prove this by contradiction.

Suppose that we can schedule the committees in *m* one-hour time periods, where $m < K$.

We can then give G_0 an *m*-colouring by colouring with the same colour all vertices which correspond to committees meeting at the same time.

To see that this is, infact, a legitimate *m*-colouring of G_0, consider two adjacent vertices.

These vertices correspond to two committees containing one or more common members.

Hence, these committees meet at different times, and thus the vertices are coloured differently.

However, an *m*-colouring of G_0 gives a contradiction since we have $\chi(G_0) = K$.

Problem 2.52. *Suppose* $\chi(G) = 1$ *for some graph G. What do you know about G ?*

Solution. If G has an edge, its end vertices must be coloured differently, so $\chi(G) \geq 2$.

Thus $\chi(G) = 1$ if and only if G has no edges.

Problem 2.53. *Any two cycles are homeomorphic. Why ?*

Solution. Any cycle can be obtained from a 3-cycle by adding vertices to edges.

Problem 2.54. *Find the number N defined in this proof for the graph of Figure (2.77). Verify that* $N \leq 2E$. *Give an example of an edge which is counted just once.*

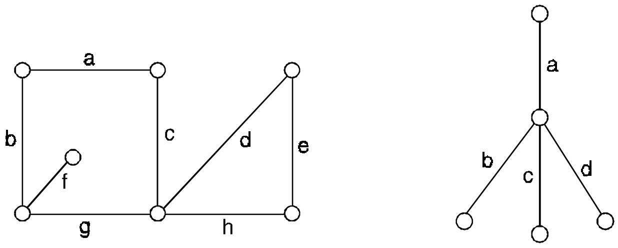


Fig. 2.77.

Solution. The boundaries of the regions are gievn :

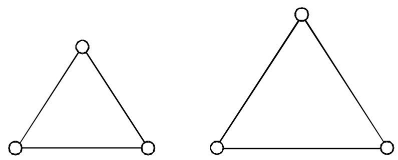
{*d*, *e*, *h*}, {*a*, *b*, *f*, *g*, *c*} and {*a*, *b*, *g*, *c*, *d*, *e*, *h*}

N = 3 + 5 + 7 = 15 ≤ 16 = 2 E.

Edge *f* is counted only once.

Problem 2.55. *Show that, Euler's theorem is not necessarily true if "connected" is omitted from its statement.*

Solution.



In the graph shown, V – E + R = 6 – 6 + 3 = 3.

Problem 2.56. *Consider the plane graph shown on the left of Figure 2.78, below :*

(*a*) *How many regions are there ?*

(*b*) *List the edges which form the boundary of each region.*

(*c*) *Which region is exterior ?*

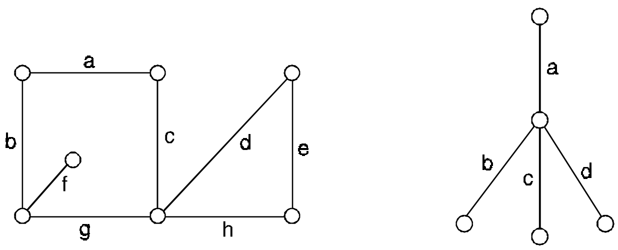


Figure 2.78.

Solution. The graph on the left of Figure 2.78 has three regions whose boundaries are {*d*, *e*, *h*}, {*a*, *b*, *f*, *g*, *c*} and {*a*, *b*, *g*, *c*, *d*, *e*, *h*}, the last region is exterior.

The graph on the right is a tree, it determines only one region, the exterior one, with boundary {*a*, *b*, *c*, *d*}.

2.16.2. Scheduling Final Exams (2.57)

How can the final exams at a university be scheduled so that no student has two exams at the same time ?

Solution. This scheduling problem can be solved using a graph model, with vertices representing courses and with an edge between two vertices if there is a common student in the courses they represent. Each time slot for a final exam is represented by a different colour. A scheduling of the exams corresponds to a colouring of the associated graph.

For instance, suppose there are seven finals to be scheduled. Suppose the courses are numbered 1 through 7. Suppose that the following pairs of courses have common students : 1 and 2, 1 and 3, 1 and 4, 1 and 7, 2 and 3, 2 and 4, 2 and 5, 2 and 7, 3 and 4, 3 and 6, 3 and 7, 4 and 5, 4 and 6, 5 and 6, 5 and 7, and 6 and 7.

In Figure 2.79, the graph associated with this set of classes is shown.

A scheduling consists of a colouring of this graph.

Since the chromatic number of this graph is 4, four times slots are needed.

A colouring of the graph using four colours and the associated schedule are shown in Figure 2.80.

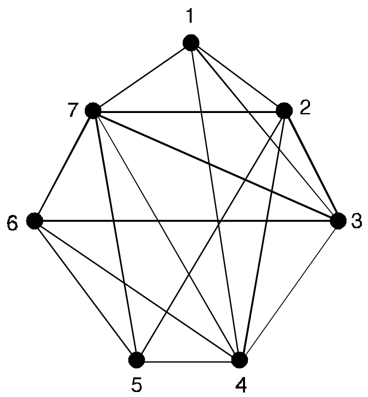


Fig. 2.79.

The graph representing the scheduling of final exams

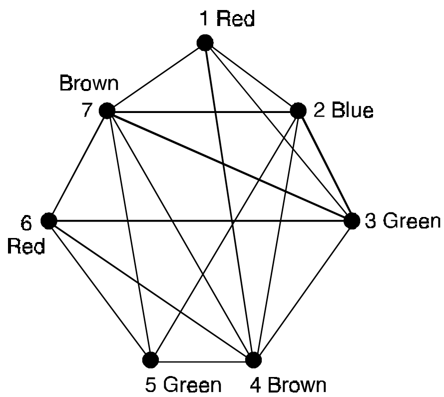


Fig. 2.80.

Using a colouring to schedule final exams.

Time period	Courses
I	1, 6
II	2
III	3, 5
IV	4, 7

2.16.3. Frequency assignments (2.58)

Television channels 2 through 13 are assigned to stations in New Delhi so that no two stations within 150 miles can operate on the same channel. How can the assignment of channels be modeled by graph colouring ?

Solution. Construct a graph by assigning a vertex to each station.

Two vertices are connected by an edge if they are located within 150 miles of each other.

An assignment of channels corresponds to a colouring of the graph. Where each colour represents a different channel.

2.16.4. Index registers (2.59)

In efficient compilers the execution of loops is speeded up when frequently used variables are stored temporarily in index registers in the central processing unit, instead of in regular memory. For a given loop, how many index registers are needed ?

Solution. This problem can be addressed using a graph colouring model.

To set up the model, let each vertex of a graph represent a variable in the loop.

There is an edge between two vertices if the variables they represent must be stored in index registers at the same time during the execution of the loop.

Thus, the chromatic number of the graph gives the number of index registers needed, since different registers must be assigned to variables when the vertices respresenting these variables are adjacent in the graph.

Problem 2.60. *What is the chromatic number of the graph C_n ?*

Solution. We will first consider some individual cases.

To begin, let $n = 6$. Pick a vertex and colour it red.

Proceed clockwise in the planar depiction of C_6 shown in Figure (2.81).

It is necessary to assign a second colour, say blue, to the next vertex reached.

Continue in the clockwise direction, the third vertex can be coloured red, the fourth vertex blue, and the fifth vertex red.

Finally, the sixth vertex, which is adjacent to the first, can be coloured blue.

Hence, the chromatic number of C_6 is 2. Figure (2.81) displays the colouring constructed here.

Next, let $n = 5$ and consider C_5. Pick a vertex and colour it red.

Proceeding clockwise, it is necessary to assign a second colour, say blue, to the next vertex reached.

Continuing in the clockwise direction, the third vertex can be coloured red, and the fourth vertex can be coloured blue.

The fifth vertex cannot be coloured either red or blue, since it is adjacent to the fourth vertex and the first vertex.

Consequently, a third colour is required for this vertex.

Note that we would have also needed three colours if we had coloured vertices in the counter clockwise direction.

Thus, the chromatic number of C_5 is 3. A colouring of C_5 using three colours is displayed in Figure (2.81).

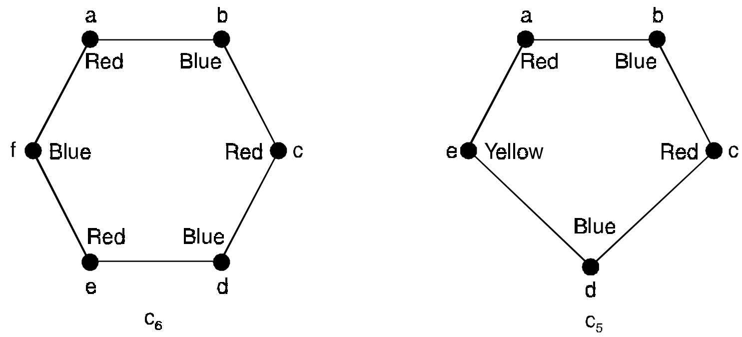


Fig. 2.81. Colourings of C_5 and C_6.

In general, two colours are needed to colours C_n when n is even. To construct such a colouring, simply pick a vertex and colour it red.

Proceeding around the graph in a clockwise direction (using a planar representation of the graph) colouring the second vertex blue, the third vertex red, and so on.

The nth vertex can be colored blue, since the two vertices adjacent to it, namely the $(n - 1)$st and the first vertices, are both coloured red.

When n is odd and $n > 1$, the chromatic number of C_n is 3.

To see this, pick an initial vertex. To use only two colours, it is necessary to alternate colours as the graph is traversed in a clockwise direction.

However, the nth vertex reached is adjacent to two vertices of different colours, namely, the first and $(n - 1)$st.

Hence, a third colour must be used.

Problem 2.61. *What is the chromatic number of the complete bipartite graph $K_{m,\,n}$, where m and n are positive integers ?*

Solution. The number of colours needed may seem to depend on m and n.

However, only two colours are needed. Colour the set of m vertices with one colour and the set of n vertices with a second colour.

Since edges connect only a vertex from the set of m vertices and a vertex from the set of n vertices, no two adjacent vertices have the same colour.

A colouring of $K_{3,\,4}$ with two colours is displayed in Figure (2.82).

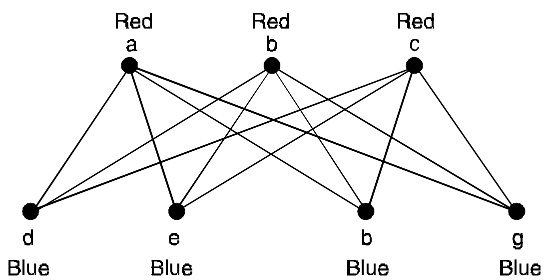


Fig. 2.82. A colouring of $K_{3,\,4}$.

Problem 2.62. *What is the chromatic number of K_n ?*

Solution. A colouring of K_n can be constructed using n colours by assigning a different color to each vertex. Is there a colouring using fewer colours ? The answer is no. No two vertices can be assigned the same colour, since every two vertices of this graph are adjacent.

Hence, the chromatic number of $K_n = n$.

A colouring of K_5 using five colours is shown in Figure (2.83).

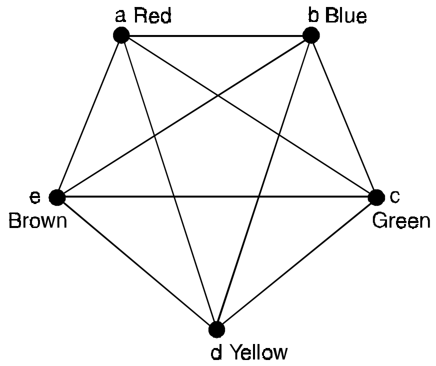


Fig. 2.83. A colouring of K_5.

Problem 2.63. *What is the chromatic numbers of the graphs G and H shown in Figure (2.84).*

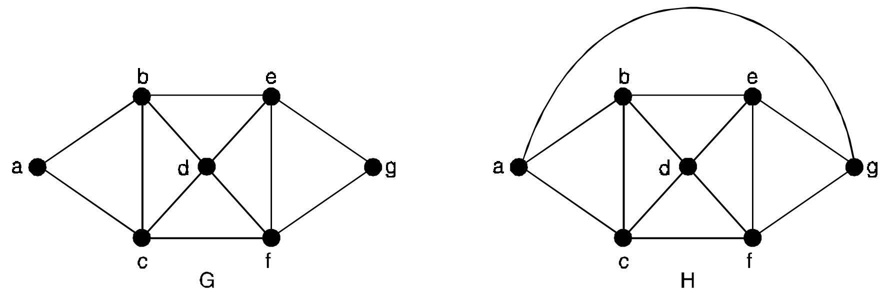


Fig. 2.84. The simple graphs G and H.

Solution. The chromatic number of G is at least three, since the vertices a, b and c must be assigned different colorus.

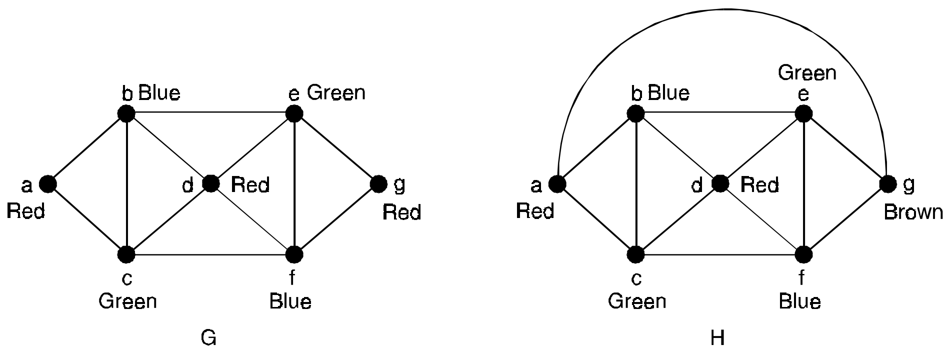


Fig. 2.85. Colourings of the graphs G and H.

To see if G can be colourd with three colours, assign red to *a*, blue to *b*, and green to *c*. Then, *d* can (and must) be coloured red since it is adjacent to *b* and *c*.

Furthermore, *e* can (and must) be coloured green since it is adjacent only to vertices coloured red and blue, and *f* can (and must) be coloured blue since it is adjacent only to vertices coloured red and green.

Finally, *g* can (and must) be coloured red since it is adjacent only to vertices coloured blue and green.

This produces a colouring of G using exactly three colours. Figure (2.85) displays such a colouring.

The graph H is made up of the graph G with an edge connecting *a* and *g*.

Any attempt to colour H using three colours must follow the same reasoning as that used to colour G, except at the last stage, when all vertices other than *g* have been coloured.

Then, since *g* is adjacent (in H) to vertices coloured red, blue, and green, a fourth colour, say brown, needs to be used.

Hence, H has a chromatic number equal to 4.

A colouring of H is shown in Figure (2.85).

Problem 2.64. *Suppose that in one particular semester, there are students taking each of the following combinations of courses.*

* *Mathematics, English, Biology, Chemistry*

* *Mathematics, English, Computer Science, Geography*

* *Biology, Psychology, Geography, Spanish*

* *Biology, Computer Science, History, French*

* *English, Psychology, History, Computer Science*

* *Psychology, Chemistry, Computer Science, French*

* *Psychology, Geography, History, Spanish.*

What is the minimum number of examination periods required for exams in the ten courses specified so that students taking any of the given combinations of courses have no conflicts ?

Find a possible schedule which uses this minimum number of periods.

Solution. In order to picture the situation, we draw a graph with ten vertices labeled M, E, B, ... corresponding to Mathematics, English, Biology and so on, and join two vertices with an edge if exams in the corresponding subjects must not be scheduled together.

The minimum number of examination periods is evidently the chromatic number of this graph. What is this ? Since the graph contains K_5 (with vertices M, E, B, G, CS), at least five different colours are needed. (The exams in the subjects which these vertices represent must be scheduled at different times). Five colours are not enough, however, since P and H are adjacent to each other and to each of E, B, G and CS.

The chromatic number of the graph is, infact 6.

In Figure (2.86), we show a 6-colouring and the corresponding exam schedule.

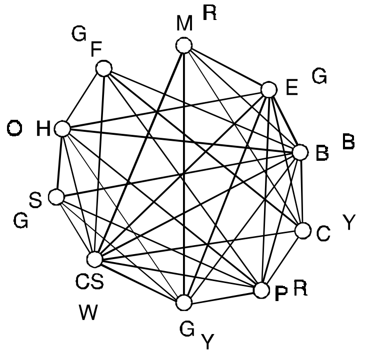


Fig. 2.86.

Period 1	Mathematics, Psychology
Period 2	English, Spanish, French
Period 3	Biology
Period 4	Chemistry, Geography
Perido 5	Computer Science
Period 6	History

Theorem 2.33. *A graph G is bipartite if and only if it does not contain a odd cycle.*

Proof. Let G be bipartite. Then the vertex set G can be partitioned into two subsets V_1 and V_2 such that every edge in G joins a vertex in V_1 with a vertex in V_2.

Suppose G contains a cycle. Let v be a vertex of this cycle. Then to trace the cycle starting from v we have to travel on the edges of G.

The edges of G are the only edges between V_1 and V_2.

Thus starting from v to come back to v along the cycle of G we have to travel exactly even number of times between V_1 and V_2.

That is, the number of edges in C is even, that is, the length of C is even.

Conversely, without loss of generality we assume G is connected.

Let G does not contain a odd cycle. Choose a vertex x of G. Colour the vertex by the Colour Black. Colour all the vertices that are at odd distances from x with the colour Red. Color all the vertices that are at even distances from x with colour Black. Since every distance is either a odd or even (but not both), every vertex of G is now coloured.

We now show that the graph G is now properly coloured. Suppose G is not properly coloured, the G contains two adjacent vertices say u and v, colored with the same colour. Then distance from the vertex x to both the vertices u and v is odd.

Let P_1 and P_2 be shortest paths from x to u and x to v respectively.

Let y be the last vertex common to P_1 and P_2 (*i.e.,* the path from y to u and path from y to v along P_1 and P_2 are disjoint). Then $d(x, y)$ along P_1 is same along P_2 (since both P_1 and P_2 are shortest paths).

Otherwise, if the $d(x, y)$ along P_1 is smaller than that on P_2, then the path from x to y along P_1 with the path from y to v along P_2 is shorter than P_2, which is a contradiction to the fact that P_2 is shortest.

Let $d(x, u) = m$ and $d(x, v) = n$, then both m and n are odd numbers or both are even numbers (since u and v are coloured with same colour).

Then $d(y, u)$ and $d(y, v)$ are both either odd or even and hence the sum is even.

Hence, the circuit formed due to these paths together with the edge uv is of odd length, which is a contradiction.

Thus we conclude that the colouring is proper.

Now consider the set V_1 of all vertices of G coloured by Black and the set V_2 of all the vertices of G coloured by the colour Red.

These sets are the partition of G such that no two vertices in the same set are adjacent.

Hence G is bipartite.

Theorem 2.34. *A graph of n vertices is a complete graph if and only if its chromatic polynomial is*

$$P_n(\lambda) = \lambda(\lambda - 1)(\lambda - 2) \ldots\ldots (\lambda - n + 1).$$

Proof. With λ colours, there are λ different ways of colouring any selected vertex of a graph.

A second vertex can be coloured properly in exactly $\lambda - 1$ ways, the third in $\lambda - 2$ ways, the fourth in $\lambda - 3$ ways, ..., and the nth in $\lambda - n + 1$ ways if and only if every vertex is adjacent to every other.

That is, if and only if the graph is complete.

Theorem 2.35. *Let a and b be two non adjacent vertices in a graph G. Let G′ be a graph obtained by adding an edge between a and b. Let G″ be a simple graph obtained from G by fusing the vertices a and b together and replacing sets of parallel edges with single edges. Then*

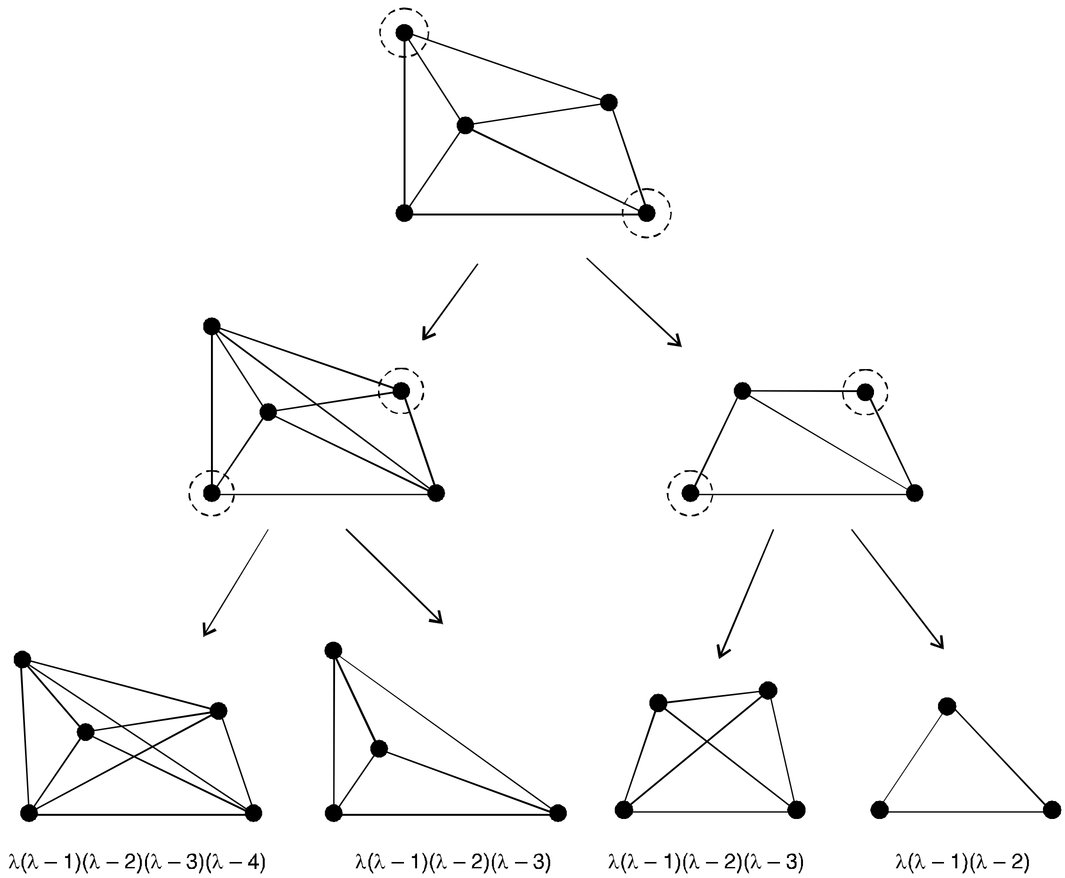
$$P_n(\lambda) \text{ of } G = P_n(\lambda) \text{ of } G' + P_{n-1}(\lambda) \text{ of } G''.$$

Proof. The number of ways of properly colouring G can be grouped into two cases, one such that vertices *a* and *b* are of the same colour and the other such that *a* and *b* are of different colours.

Since the number of ways of properly colouring G such that *a* and *b* have different colours = number of ways of properly colouring G′, and

Number of ways of properly colouring G such that *a* and *b* have the same colour = number of ways of properly colouring G″.

$$P_n(\lambda) \text{ of } G = P_n(\lambda) \text{ of } G' + P_{n-1}(\lambda) \text{ of } G''$$



$$P_5(\lambda) \text{ of } G = \lambda(\lambda-1)(\lambda-2) + 2\lambda(\lambda-1)(\lambda-2)(\lambda-3) + \lambda(\lambda-1)(\lambda-2)(\lambda-3)(\lambda-4)$$
$$= \lambda(\lambda-1)(\lambda-2)(\lambda^2-5\lambda+7)$$

Fig. 2.87. Evaluation of a chromatic polynomial.

Theorem 2.36. *A graph is bicolourable if and only if it has no odd cycles.*

Theorem 2.37. *For any graph G, $\chi(G) \leq 1 + \max \delta(G')$,*

Where the maximum is taken over all induced subgraphs G′ of G.

Proof. The result is obvious for totally disconnected graphs.

Let G be an arbitrary n-chromatic graph, $n \geq 2$.

Let H be any smallest induced subgraph such that $\chi(H) = n$

The graph H therefore has the property that

$$\chi(H - v) = n - 1 \text{ for all its points } v.$$

It follows that deg $v \geq n - 1$ so that $\delta(H) \geq n - 1$ and hence

$$n - 1 \leq \delta(H) \leq \max \delta(H') \leq \max \delta(G')$$

The first maximum taken over all induced subgraphs H′ of H and the second over all induced subgraphs G′ of G.

This implies that

$$\chi(G) = n < 1 + \max \delta(G')$$

Corollary : For any graph G, the chromatic number is atmost one greater than the maximum degree $\chi \leq 1 + \Delta$.

Theorem 2.38. If $\Delta(G) = n \geq 2$, then G is n-colourable unless, or

(*i*) $n = 2$ and G has a component which is an odd cycle, or

(*ii*) $n > 2$ and K_{n+1} is a component of G.

Theorem 2.39. *For any graph G,* $\dfrac{P}{\beta_0} \leq \chi \leq P - \beta_0 + 1$.

Proof. If $\chi(G) = n$, then V can be partitioned into n colour classes $V_1, V_2, \ldots\ldots V_n$, each of which, as noted above, is an independent set of points.

If $|V_i| = P_i$, then every $P_i \leq \beta_0$ so that

$$P = \Sigma P_i \leq n\, \beta_0$$

To verify the upper bound, let S be a maximal independent set containing β_0 points.

It is clear that $\chi(G - S) \geq \chi(G) - 1$.

Sicne G – S has $P - \beta_0$ points, $\chi(G - S) \leq P - \beta_0$

Therefore, $\chi(G) \leq \chi(G - S) + 1 \leq P - \beta_0 + 1$.

Theorem 2.40. *For every two positive integers m and n, there exists an n-chromatic graph whose girth exceeds m.*

Theorem 2.41. *For any graph G, the sum and product of χ and $\bar{\chi}$ satisfy the inequalities :*

$$2\sqrt{P} \leq \chi + \bar{\chi} \leq P + 1,$$

$$P \leq \chi\bar{\chi} \leq \left(\frac{P+1}{2}\right)^2$$

Proof. Let G be n-chromatic and let $V_1, V_2, \ldots V_n$, be the colour classes of G, where $|V_i| = P_i$

Then of course $\Sigma P_i = P$ and max $P_i \geq \frac{P}{n}$.

Since each V_i induces a complete subgraph of $\overline{G}$

$\bar{\chi} \geq \max P_i \geq \frac{P}{n}$ so that $\chi\bar{\chi} \geq P$.

Since the geometric mean, it follows that $\chi + \bar{\chi} \geq 2\sqrt{P}$.
This establishes both lower bounds.

To show that $\chi + \bar{\chi} \leq P + 1$, we use induction on P, noting that equality holds when P = 1.

We thus assume that $\chi(G) + \bar{\chi}(G) \leq P$ for all graphs G having P – 1 points.

Let H and $\overline{H}$ be complementary graphs with P points, and let v be a point of H.

Then $G = H - v$ and $\overline{G} + \overline{H} - v$ are complementary graphs with P – 1 points.

Let the degree of v in H be d so that the degree of v in $\overline{H}$ is $P - d - 1$.
It is obvious that

$$\chi(H) \leq \chi(G) + 1 \text{ and } \bar{\chi}(H) \leq \bar{\chi}(G) + 1$$

If either

$$\chi(H) < \chi(G) + 1 \text{ or } \bar{\chi}(H) < \bar{\chi}(G) + 1.$$

then $\chi(H) + \bar{\chi}(H) \leq P + 1$.

Suppose then that $\chi(H) = \chi(G) + 1$ and $\bar{\chi}(H) = \bar{\chi}(G) + 1$.

This implies that the removal of v from H, producing G, decreases the chromatic number so that $d \geq \chi(G)$.

Similarly $P - d - 1 \geq \bar{\chi}(G)$,

thus $\chi(G) + \bar{\chi}(G) \leq P - 1$

Therefore, we always have $\chi(H) + \bar{\chi}(H) \leq P + 1$
Finally, applying the inequality

$$4\chi\bar{\chi} \leq (\chi + \bar{\chi})^2 \quad \text{we see that} \quad \chi\bar{\chi} \leq \left[\frac{(P+1)}{2}\right]^2.$$

Theorem 2.42. *Every tree T with two or more vertices is 2-chromatic.*

Proof. Since Tree T is a bipartite graph.

The vertex set V of G can be partitioned into two subsets V_1 and V_2 such that no two vertices of the set V_1 are adjacent and two vertices of the set V_2 are adjacent.

Now colour the vertices of the set V_1 by the colour 1 and the vertices of the set V_2 by the colour 2.

This colouring is a proper colouring.

Hence, chromatic number of G ≤ 2, and since T contains atleast one edge chromatic number of G ≥ 2.

Thus, chromatic number of G is 2.

Theorem 2.43. *A graph G is 2-chroamtic if and only if G is bipartite.*

Proof. Let chromatic index of a graph G be two.

Let G be properly coloured with two colours 1 and 2. Consider the set of vertices coloured with the colour 1 and the set of all vertices coloured with the colour 2.

These sets are precisely partition of the vertex set such that no two of the vertices of the same set are adjacent.

Hence G is bipartite.

Conversely, G is not bipartite then G contains a odd cycle.

The chromatic number of a odd cycle is three.

Hence G contains a subgraph whose chromatic number is three.

Therefore, K(G) ≥ 3.

Theorem 2.44. *The chromatic number of a graph cannnot exceed one more than the maximum degree of a vertex of G.*

Proof. Since maximum degree of the graph is m, the graph cannot have a subgraph K_n, $n > m + 1$.

Thus K(G) $\leq m + 1$.

Corollary. The chromatic number of a graph cannot exceed maximum degree m of a vertex of G if and only if G does not have a subgraph isomorphic to K_{m+1}.

Theorem 2.45. *If d_{max} is the maximum degree of the vertices in a graph G, chromatic number of G $\leq 1 + d_{max}$.*

Theorem 2.46. (König's theorem)

A graph with atleast one edge is 2-chromatic if and only if it has no circuits of odd length.

Proof. Let G be a connected graph with circuits of only even lengths.

Consider a spanning tree T in G, let us properly color T with two colors. Now add the chords to T one by one.

Since G had no circuits of odd length, the end vertices of every chord being replaced are differently coloured in T.

Thus G is coloured with two colours, with no adjacent vertices having the same colour.

That is, G is 2-chromatic.

Conversely, if G has a circuit of odd length, we would need at least three colours just for that circuit.

Thus the theorem.

Theorem 2.47. *A graph G is 2-chromatic if and only if it is a non-null bipartite graph.*

Proof. Suppose a graph G is 2-chromatic. Then it is non-null, and some vertices of G have one colour, say α, and the rest of the vertices have another colour, say β.

Let V_1 be the set of vertices having colour α and V_2 be the set of vertices having colour β.

Then $V_1 \cup V_2 = V$, the vertex set of G, and $V_1 \cap V_2 = \phi$.

Also, no two vertices of V_1 can be adjacent and no two vertices of V_2 can be adjacent.

As such, every edge in G has one end in V_1 and the other end in V_2.

Hence G is a bipartite graph.

Conversely, suppose G is a non-null bipartite graph. Then the vertex set of G has two partitions V_1 and V_2 such that every edge in G has one end in V_1 and another end in V_2.

Consequently, G cannot be properly coloured with one colour, because then vertices in V_1 and V_2 will have the same colour and every edge has both of its ends of the same colour.

Suppose we assign a colour α to all vertices in V_1 and a different colour β to all vertices in V_2.

This will make a proper colouring of V.

Hence G is 2-chromatic.

Corollary. *Every three with two or more vertices is a bipartite graph.*

Proof. Every tree with two or more vertices is 2-chromatic. Therefore, it is bipartite, by the theorem.

Theorem 2.48. For a graph G, the following statements are equivalent :

(*i*) G is 2-chromatic

(*ii*) G is non-null and bipartite

(*iii*) G has no circuits of odd length.

Corollary. A graph G is a non-null bipartite graph if and only if it has no circuits of odd length.

Theorem 2.49. *If G is a graph with n vertices and degree* δ*, then* $\chi(G) \geq \dfrac{n}{n-\delta}$.

Proof. Recall that δ is the minimum of the degrees of vertices.

Therefore, every vertex v of G has atleast δ number of vertices adjacent to it.

Hence there are at most $n - \delta$ vertices can have the same colour.

Let K be the least number of colours with which G can be properly coloured.

Then $K = \chi(G)$.

Let $\alpha_1, \alpha_2, ... \alpha_K$ be these colours and let n_1 be the number of vertices having colour α_1, n_2 be the number of vertices having colour α_2 and so on, and finally n_K be the number of vertices having colour α_K.

Then $$n_1 + n_2 + n_3 + ... n_k = n \qquad ...(1)$$

and $$n_1 \leq n - \delta,\ n_2 \leq n - \delta,\ ......\ n_k \leq n - \delta \qquad ...(2)$$

Adding the K in equalities in (2), we obtain

$$n_1 + n_2 + ... + n_k \leq K(n - \delta)$$

or $$n \leq K(n - \delta), \text{ using (1)}$$

Since $K = \chi(G)$, this becomes

$$\chi(G) \geq \frac{n}{n-\delta}$$

This is the required result.

Problem 2.65. *Write down chromatic polynomial of a given graph on n vertices.*

Solution. Let G be a graph on n vertices.

Let C_i denote the different ways of properly coloring of G using exactly i distinct colors.

These i colors can be chosen out of λ colors in $\binom{\lambda}{i}$ distinct ways.

Thus total number of distinct ways a proper coloring to a graph with i colors out of λ colors is possible in $\binom{\lambda}{i}C_i$ ways.

Hence $\sum_{i=1}^{n} \binom{\lambda}{i}C_i$. Each C_i has to be evaluated individually for the given graph.

Problem 2.66. *Find all maximal independent sets of the following graph.*

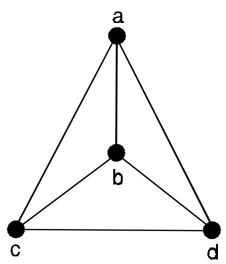


Fig. 2.88.

Solution. The maximal independent sets of G are $\{a\}$, $\{b\}$, $\{c\}$ and $\{d\}$.

Problem 2.67. *Find all maximal independent sets of the following graph.*

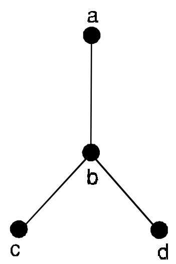


Fig. 2.89.

Solution. Maximal independent sets are $\{a, c, d\}$ and $\{b\}$.

Problem 2.68. *Find all possible maximal independent sets of the following graph using Boolean expression.*

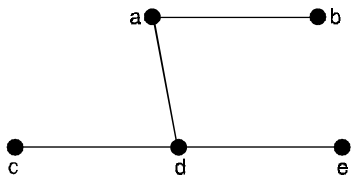


Fig. 2.90.

Solution. The Boolean expression for this graph

$$\phi = \Sigma xy = ab + ad + cd + de \text{ and}$$

$$\begin{aligned}\phi' &= (a' + b')(a' + d')(c' + d')(d' + e')\\ &= \{a'(a' + d') + b'(a' + d')\}\ \{c'(d' + e') + d'(d' + e')\}\\ &= \{a' + b'a' + b'd'\}\ \{c'd' + c'e' + d'\}\\ &= \{a'(1 + b') + b'd'\}\ \{d'(c' + 1) + c'e'\}\\ &= \{a' + b'd'\}\ \{d' + c'e'\}\\ &= a'd' + a'c'e' + b'd' + b'c'd'e'\\ &= a'd' + a'c'e' + b'd'(1 + c'e')\\ &= a'd' + a'c'e' + b'd'\end{aligned}$$

Thus $f_1 = a'd'$, $f_2 = a'c'e'$ and $f_3 = b'd'$.

Hence maximal independent sets are $V - \{a, b\} = \{b, c, e\}$

$V - \{a, c, e\} = \{b, d\}$ and $V - \{b, d\} = \{a, c, e\}$.

Problem 2.69. *Find the chromatic polynomial of a connected graph on three vertices.*

Solution. Since the graph is connected it contains an edge, hence minimum two colours are required for any proper colouring of G.

Thus $C_1 = 0$.

Further the number of ways a graph on n vertices with n distinct colours can be properly assigned in n ! ways.

Hence for the graph on 3 vertices $C_3 = 3 ! = 6$.

If G is a triangle, then G cannot be labeled with two colours.

Hence $C_2 = 0$, thus

$$P_3(\lambda) = \sum_{i=1}^{3} \binom{\lambda}{i} C_i = 0 + 0 + \binom{\lambda}{3} 6$$

$$= \frac{\lambda(\lambda-1)(\lambda-2)}{3!} 6 = \lambda(\lambda - 1)(\lambda - 2)$$

If G is a path, then end vertices can be coloured with only two ways with two colours and for each choice of end vertex only one choice of another colour is possible for the middle vertex. Thus $C_2 = 2$ and similar to above argument $C_3 = 3 !$.

Therefore, $P_3(\lambda) = \sum_{i=1}^{3} \binom{\lambda}{i} C_i = 0 + \binom{\lambda}{2} 2 + \binom{\lambda}{3} 6$

$$= \frac{\lambda(\lambda-1)}{2!} 2 + \frac{\lambda(\lambda-1)(\lambda-2)}{3!} 6$$

$$= \lambda(\lambda - 1) + \lambda(\lambda - 1)(\lambda - 2)$$

$$= \lambda(\lambda - 1)(1 + (\lambda - 2))$$

$$= \lambda(\lambda - 1)^2.$$

Theorem 2.50. *An n-vertex graph is a tree if and only if its chromatic polynomial* $P_n(\lambda) = \lambda(\lambda - 1)^{n-1}$.

Proof. Let G be a tree on n vertices.

We prove the result by induction on n.

If $n = 1$, then G contains only one vertex which can be coloured in λ distinct ways only.

Hence the result holds in this case.

If $n = 2$, then G contains one edge, so that exactly two colours are required for the proper colouring of the graph.

Hence $C_1 = 0$ and two colours can be assigned in two different ways for the vertices of the graph.

Therefore, $C_2 = 2$.

Thus $P_n(\lambda) = 0 + \left[\frac{\lambda(\lambda-1)}{2!}\right] 2 = \lambda(\lambda - 1)$

Hence the result holds with $n = 2$.

Now assume the result for lesser values of n, $n \geq 2$.

Since the graph G is a tree, it contains a pendent vertex. Let v be a pendent vertex of the graph. Let G′ be the graph obtained by deleting the vertex v. Then by inductive hypothesis the chromatic polynomial of G′ is $\lambda(\lambda - 1)^{n-2}$.

Now for each proper coloring of G′ the given graph can be properly colored by painting the vertex v with the colour other than vertex adjacent to the vertex v.

Thus we can choose $(\lambda - 1)$ colors to v for each proper colouring of G′.

Hence total $\lambda(\lambda - 1)^{n-2} (\lambda - 1) = \lambda(\lambda - 1)^{n-1}$ ways we can properly colour the given tree.

Thus the result hold by induction.

Problem 2.70. *How many ways a tree on 5 vertices can be properly coloured with at most 4 colors.*

Solution. We have a tree with n vertices can be coloured with at most λ colours in $\lambda(\lambda - 1)^{n-1}$ ways.

Therefore a tree on $n = 5$ vertices can be properly coloured with at most $\lambda = 4$ colours in $\lambda(\lambda - 1)^{n-1}$ $= 4 \cdot 3^4 = 4 \cdot 81 = 324$ ways.

Problem 2.71. *Write down the chromatic polynomial of the graph $K_4 - e$.*

Solution.

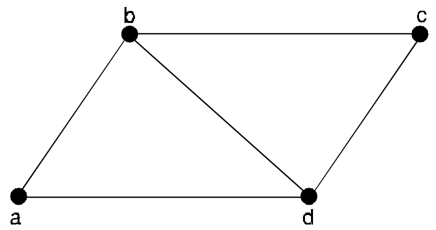


Fig. 2.91.

The graph $K_4 - e$ is shown below. It contains exactly two non-adjacent vertices.

Let G′ be a graph obtained by adding the edge between these non adjacent vertices.

Then G′ is a complete graph K_4.

Hence $P_4(\lambda)$ of $G' = \lambda(\lambda - 1)(\lambda - 2)(\lambda - 3)$

Let G″ be the graph obtained by fusing these vertices and replacing the parallel edges.

Then G″ is a complete graph K_3.

Hence $P_3(\lambda)$ of $G'' = \lambda(\lambda - 1)(\lambda - 2)$

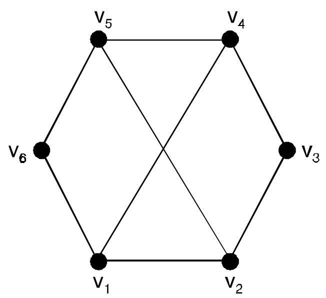
Now, $P_4(\lambda)$ of $G = P_4(\lambda)$ of $G' + P_{4-1}(\lambda)$ of G''

$$= \lambda(\lambda - 1)(\lambda - 2)(\lambda - 3) + \lambda(\lambda - 1)(\lambda - 2)$$

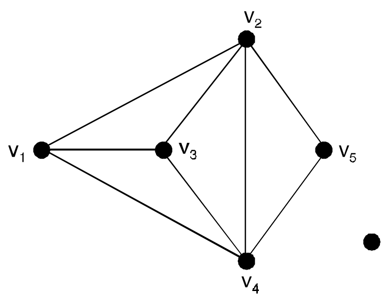
$$= \lambda(\lambda - 1)(\lambda - 2)(1 + \lambda - 3)$$

$$= \lambda(\lambda - 1)(\lambda - 2)^2.$$

Problem 2.72. *Find the chromatic number of the following graphs*



(*a*)



(*b*)

Fig. 2.92.

Solution. (*i*) For the graph in Figure 2.92(*a*), let us assign a colour α to the vertex v_1.

Then, for a proper colouring, we have to assign a different colour to its neighbours v_2, v_4, v_6.

Since v_2, v_4, v_6 are non adjacent vertices, they can have the same color, say β (which is different from α).

Since v_3, v_5 are not adjacent to v_1, these can have the same colours as v_1, namely α.

Thus, the graph can be properly coloured with at least two colours, with the vertices v_1, v_3, v_5 having one colour α and v_2, v_4, v_6 having a different colour β.

Hence the graph is 2-chromatic

(*i.e.*, the chromatic number of the graph is 2).

(*ii*) For the graph in Figure 2.92(*b*), let us again the colour α to the vertex v_1.

Then, for a proper colouring, its neighbours v_2, v_3 and v_4 cannot have the colour α, but v_5 can have the colour α.

Further more, v_2, v_3, v_4 must have different colours, say β, γ, δ.

Thus, at least four colours are required for a proper colouring of the graph.

Hence, the graph is 4-chromatic (*i.e.*, the chroamtic number of the graph is 4).

Problem 2.73. *Prove that a simple planar graph G with less than 30 edges in 4-colorable.*

Solution. If G has 4 or less number of vertices, the required result is true.

Assume that the result is true for any graph with n = K vertices.

Consider a graph G′ with K + 1 vertices and less than 30 edges.

Then, G′ has at least one vertex v of degree at most 4.

Now, considering the graph G′ – v we find that G′ is 4-colorable.

Problem 2.74. *Prove that a graph of order n (≥ 2) consisting of a single circuit is 2-chromatic if n is even, and 3-chromatic if n is odd.*

Solution. The given graph is the cycle graph C_n, $n \geq 2$ as shown in figure below.

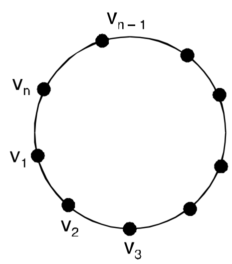


Fig. 2.93.

Obviously, the graph cannot be properly colored with a single colour. Assign two colours alternatively to the vertices, starting with v_1.

That is, the odd vertices v_1, v_3, v_5 etc, will have a colour α and the even vertices v_2, v_4, v_6 etc., will have a different colour β.

Suppose n is even. Then the vertex v_n is an even vertex and therefore will have the colour β, and the graph gets properly coloured.

Therefore, the graph is 2-chormatic.

Suppose n is odd. Then the vertex v_n is an odd vertex and therefore will have the colour α, and the graph is not properly coloured. To make it properly coloured. v_n should be assigned a third colour γ. Thus, in this case, the graph is 3-chromatic.

Problem 2.75. *Prove that every tree with two or more vertices is 2-chromatic.*

Solution. Consider a tree T rooted at a vertex *v* as shown in figure below. Assign a colour α to *v* and a different colour β to all vertices adjacent to *v*. Then the vertices adjacent to those which have the color β are not adjacent to *v* (because a tree has no circuits) and are at a distance 2 from *v*. Assign the colour α to these vertices. Repeat the process until all vertices are coloured.

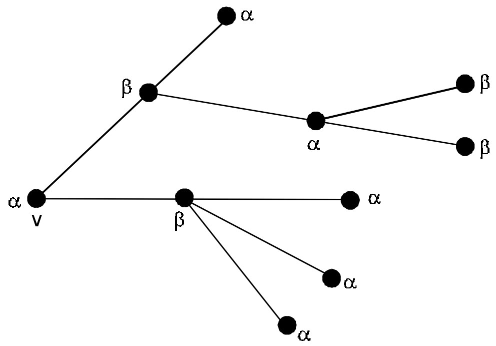


Fig. 2.94.

Thus, *v* and all vertices which are at distances 2, 4, 6, ... from *v* have α as their color and all vertices which are at distances 1, 3, 5, ... from *v* have β as their colour.

Accordingly, along any path of T the vertices are of alternating colours.

Since there is one and only one path between any two vertices in a tree, no two adjacent vertices will have the same colour.

Thus, T has been properly coloured with 2 colours.

If T has two or more vertices, it has one or more edges. As such, it cannot be coloured with 1 colour. This proves that the chromatic number of T is 2, that is 2-chromatic.

Problem 2.76. *Find the chromatic number of a cubic graph with $p \geq 6$ vertices.*

Solution. Every cubic graph contains of odd degree and in which there exists at least one triangle.

Hence $\chi(G) = 3$, where G is a cubic graph.

The following Figure (2.95) gives the result :

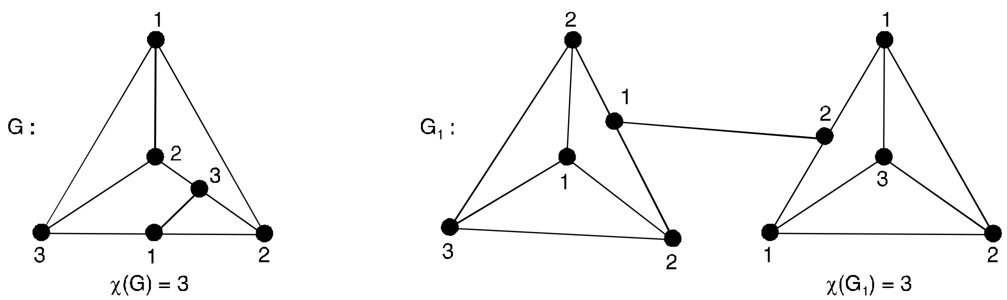


Fig. 2.95

Problem 2.77. *Find the chromatic polynomial of a complete graph on n vertices.*

Solution. Since minimum n colours required for the proper colouring of complete graph K_n on n vertices.

We have $\quad C_i = 0$ for all $i = 1, 2, \ldots\ldots n - 1$.

Further since the graph contains n vertices, n distinct colours can be assigned in $n\,!$ ways.

Thus $C_n = n\,!$.

$$\text{Therefore,} \quad P_n(\lambda) = \sum_{i=1}^{n} \binom{\lambda}{i} C_i = \binom{\lambda}{n} C_n$$

$$= \frac{\lambda(\lambda-1)(\lambda-2)\ldots\ldots\lambda-(n+1)}{n\,!}\, n\,!$$

$$= \lambda(\lambda - 1)(\lambda - 2) \ldots\ldots (\lambda - n + 1).$$

Problem 2.78. *Show that the chromatic number of a graph G is $\lambda(\lambda - 1)(\lambda - 2) \ldots\ldots (\lambda - n + 1)$ if and only if G is a complete graph on n vertices.*

Solution. For a given λ, the first vertex of a graph can be colored in λ ways.

A second vertex can be coloured properly with $\lambda - 1$ ways, the third vertex in only $\lambda - 2$ ways if and only if this vertex is adjacent to first two vertices. Continuing like this we have, the last vertex can be coloured with $(\lambda - n + 1)$ ways if and only if the graph is complete.

Problem 2.79. *Prove that, for a graph G with n vertices*

$$\beta(G) \geq \frac{n}{\chi(G)}.$$

Solution. Let K be the minimum number of colours with which G can be properly colored.

Then $K = \chi(G)$. Let $\alpha_1, \alpha_2, \ldots\ldots \alpha_K$ be these colours and let $n_1, n_2, \ldots\ldots n_K$ be the number of vertices having colours $\alpha_1, \alpha_2, \ldots\ldots \alpha_K$ respectively.

Then $n_1, n_2, \ldots\ldots n_K$ are the orders of the maximal independent sets, because a set of all vertices having the same colour contain all vertices which are mutually non-adjacent.

Sicne $\beta(G)$ is the order of a maximal independent set with largest number of vertices, none of $n_1, n_2, \ldots\ldots n_K$ can exceed $\beta(G)$.

i.e., $\quad n_1 \leq \beta(G), \quad n_2 \leq \beta(G), \ldots\ldots n_K \leq \beta(G)$

Adding these inequalities, we get

$$n_1 + n_2 + \ldots\ldots + n_K \leq K\beta(G)$$

Since $n_1 + n_2 + \ldots\ldots + n_K = n$ and $K = \chi(G)$, this becomes

$$n \leq \chi(G) \,.\, \beta(G)$$

or

$$\beta(G) \geq \frac{n}{\chi(G)}.$$

Problem 2.80. *Show that the following graph is uniquely colourable.*

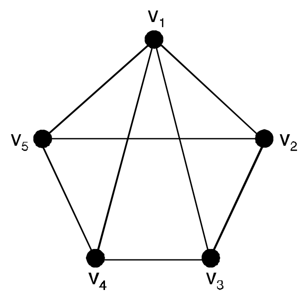


Fig. 2.96.

Solution. We check that the given graph G has only the following independent sets both of which are maximal.

$$W_1 = \{v_2, v_4\}, W_2 = \{v_3, v_5\}$$

Both of these have 2 vertices, and as such $\beta(G) = 2$.

The sets W_1 and W_2 are mutually disjoint and yield only one chromatic partition given below :

$$P = \{W_1, W_2, \{v_1\}\}$$

In view of this single possible chromatic partitioning of G, we infer that G is uniquely colourable.

2.17 COLOUR PROBLEM

The most famous unsolved problem in graph theory and perhaps in all of Mathematics is the celebrated four colour conjecture. This remarkable problem can be explained in five minutes by any mathematician to the socalled man in the steet. At the end of the explanation, both will understand the problem, but neither will be able to solve it.

The conjecture states that, any map on a plane or the surface of a sphere can be coloured with only four colours so that no two adjacent countries have the same colour. Each country must consist of a single connected region, and ajdacent countries are those having a boundary line in common. The conjecture has acted as a catalyst in the branch of mathematics known as combinatorial topology and is closely related to the currently fashionable field of graph theory. More than half a century of work by many mathematicians has yielded proofs for special cases ..... . The consensus is that the conjecture is correct but unlikely to be proved in general.

It seems destined to retain for some time the distinction of being both the simplest and most fascinating unsolved problem of mathematics.

The four colour conjecture has an interesting history, but its origin remains some what vague. There have been reports that Möbius was familiar with this problem in 1840, but it is only definite that the problem was communicated to De Morgan by Guthrie about 1850.

The first of many erroneous proofs of the conjecture was given in 1879 by Kempe. An error was found in 1890 by Heawood who showed, however, that the conjecture becomes true when ‘four’ is replaced by ‘five’.

A counter example, if ever found, will necessarily be extremely large and complicated, for the conjecture was proved most recently by Ore and Stemple for all maps with fewer than 40 countries.

The four colour conjecture is a problem in graph theory because every map yields a graph is which the countries are the points, and two points are joined by a line whenever the corresponding countries are adjacent. Such a graph obviously can be drawn in the plane without intersecting lines.

Thus, if it is possible to colour the points of every **planar graph** with four or fewer colours so that adjacent points have different colours, then the four colour conjecture will have been proved.

2.17.1. The Four colour theorem : 2.51

Every planar graph is 4-colorable.

Assume the four colour conjecture holds and let G be any plane map.

Let G* be the underlying graph of the geometric dual of G.

Since two regions of G are adjacent if and only if the corresponding vertices of G* are adjacent, map G is 4-colorable because graph G* is 4-colorable.

Conversely, assume that every plane map is 4-colorable and let H be any planar graph.

Without loss of generality, we suppose H is a connected plane graph.

Let H* be the dual of H, so drawn that each region of H* encloses precisely one vertex of H. The connected plane pseudograph H* can be converted into a plane graph H′ by introducing two vertices into each loop of H* and adding a new vertex into each edge in a set of multiple edges.

The 4-colorability of H′ now implies that H is 4-colorable, completing the verification of the equivalence.

If the four color conjecture is ever proved, the result will be best possible, for it is easy to give examples of planar graphs which are 4-chromatic, such as K_4 and W_6 (see Figure 2.97 below).

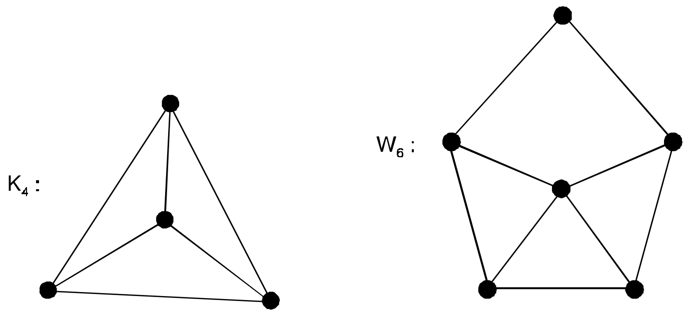


Fig. 2.97. Two 4-chromatic planar graphs.

Theorem 2.52. *Every planar graph with fewer than 4 triangles is 3-colourable.*

Corollary. Every planar graph without triangle is 3-colourable.

Theorem 2.53. *The four colour conjecture holds if and only if every cubic bridgeless plane map is 4-colourable.*

Proof. We have already seen that every plane map is 4-colourable if and only if the four colour conjecture holds.

This is also equivalent to the statement that every bridgeless plane map is 4-colourable since the elementary contraction of identifying the end vertices of a bridge affects neither the number of regions in the map nor the adjacency of any of the regions.

Certainly, if every bridgeless plane map is 4-colorable, then every cubic bridgeless plane map is 4-colorable.

In order to verify the converse, let G be a bridgeless plane map and assume all cubic bridgeless plane maps are 4-colourable.

Since G is bridgeless, it has no end vertices.

If G contains a vertex v of degree 2 incident with edges y and z, we subdivide y and z, denoting the subdivision vertices by u and w respectively.

We now remove v, identify u with one of the vertices of degree 2 in a copy of the graph $K_4 - x$ and identify w with the other vertex of degree 2 in $K_4 - x$.

Observe that each new vertex added has degree 3 (see Figure 2.98).

If G contains a vertex v_0 of degree $n \geq 4$ incident with edges x_1, x_2, ... x_n, arranged cyclically about v_0, we subdivide each x_i producing a new vertex v_i.

We then remove v_0 and add the new edges v_1v_2, v_2v_3, ..., $v_{n-1}v_n$, v_nv_1.

Again each of the vertices so added has degree 3.

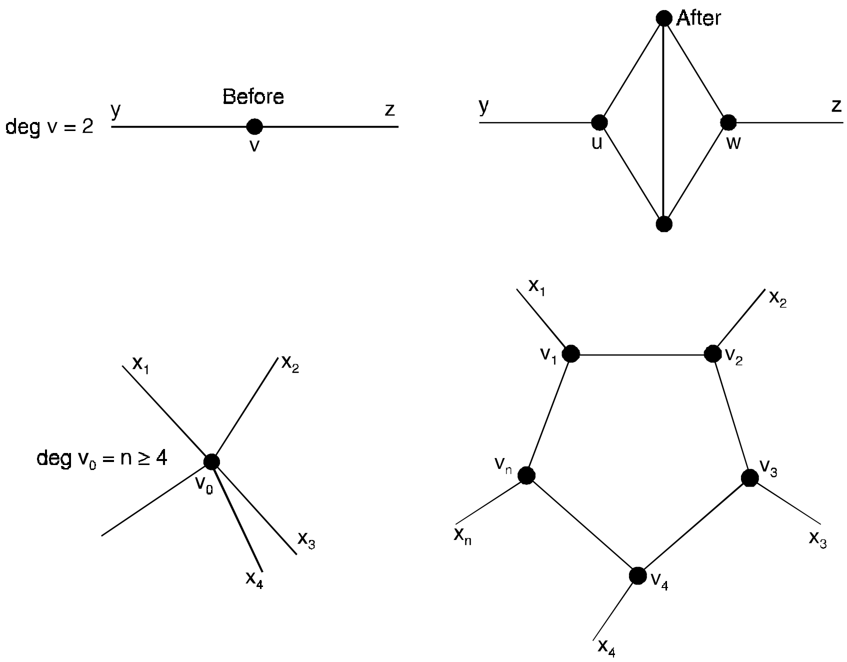


Fig. 2.98. Conversion of a graph into a cubic graph.

Denote the resulting bridgeless cubic plane map by G′, which, by hypothesis, is 4-colourable.

If for each vertex v of G with deg $v \neq 3$, we identify all the newly added vertices associated with v in the formation of G′, we arrive at G once again. Thus, let there be given a 4-colouring of G′. The above mentioned contradiction of G′ into G induces an m-colouring of G, $m \leq 4$, which completes the proof.

Theorem 2.54. *The four color conjecture holds if and only if every hamiltonian planar graph is 4-colorable.*

Theorem 2.55. *For any graph G, the line chromatic number satisfies the inequalties*

$$\Delta \leq \chi' \leq \Delta + 1.$$

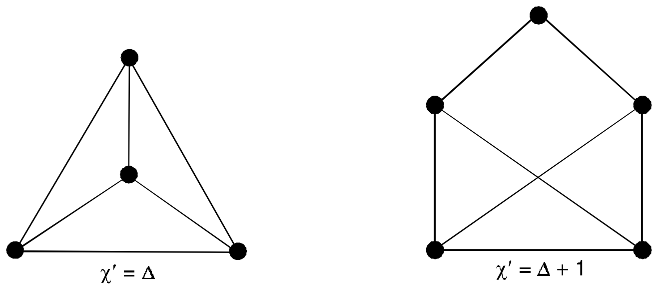


Fig. 2.99. The two values for the line-chromatic number.

2.17.2. The Five colour theorem 2.56

Every planar graph is 5-colorable.

Proof. We proceed by induction on the number P of points. For any planar graph having P ≤ 5 points, the result follows trivially since the graph is P-colorable.

As the inductive hypothesis we assume that all planar graphs with P points, P ≥ 5, are 5-colourable.

Let G be a plane graph with P + 1 vertices, G contains a vertex v of degree 5 or less.

By hypothesis, the plane graph G – v is 5-colourable.

Consider an assignment of colours to the vertices of G – v so that a 5-colouring results, when the colours are denoted by C_i, $1 \leq i \leq 5$.

Certainly, if some colour, say C_j, is not used in the colouring of the vertices adjacent with v, then by assigning the colour C_j to v, a 5-colouring of G results.

This leaves only the case to consider in which deg $v = 5$ and five colours are used for the vertices of G adjacent with v.

Permute the colours, if necessary, so that the vertices coloured C_1, C_2, C_3, C_4 and C_5 are arranged cyclically about v,

Now label the vertex adjacent with v and coloured C_i by v_i, $1 \leq i \leq 5$ (see Figure 2.100)

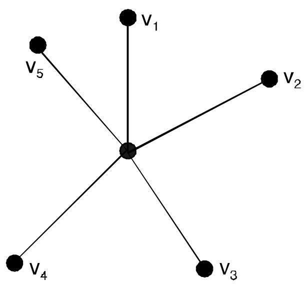


Fig. 2.100. A step in the proof of the five colour theorem.

Let G_{13} denote the subgraph of $G - v$ induced by those vertices coloured C_1 or C_3.

If v_1 and v_3 belong to different components of G_{13}, then a 5-coloring of $G - v$ may be accomplished by interchanging the colors of the vertices in the component of G_{13} containing v_1.

In this 5-coloring however, no vertex adjacent with v is colored C_1, so by coloring v with the color C_1, a 5-coloring of G results.

If, on the other hand, v_1 and v_3 belong to the same component of G_{13}, then there exists in G a path between v_1 and v_3 all of whose vertices are colored C_1 or C_3.

This path together with the path $v_1\, vv_3$ produces a cycle which necessarily encloses the vertex v_2 or both the vertices v_4 and v_5.

In any case, there exists no path joining v_2 and v_4, all of whose vertices are coloured C_2 or C_4.

Hence, if we let G_{24} denote the subgraph of $G - v$ induced by the vertices coloured C_2 or C_4, then v_2 and v_4 belong to different components of G_{24}.

Thus if we interchange colors of the vertices in the component of G_{24} containing v_2, a 5-colouring of $G - v$ is produced in which no vertex adjacent with v is coloured C_2.

We may then obtain a 5-coloring of G by assigning to v the colour C_2.

Problem Set 2.1

1. (*a*) Show that the graph is planar by drawing an isomorphic plane graph with straight edges.

(*b*) Label the regions defined by your plane graph and list the edges which form the boundary of each region.

(*c*) Verify that $V - E + R = 2$, $N \leq 2E$, and $E \leq 3V - 6$.

2. Verify Euler's formula $V - E + F = 2$ for each of the five platanoic solids.

3. If G a connected plane graph with $V \geq 3$ vertices and R regions, show that $R \leq 2V - 4$.

4. (*a*) Give an example of a connected planar graph for which $E = 3V - 6$.

(*b*) Let G be a connected plane graph for which $E = 3V - 6$ show that every region of G is a triangle.

5. (*a*) If G is a connected plane graph with at least three vertices such that no boundary of a region is a triangle, prove that $E \leq 2V - 4$.

(*b*) Let G be a connected planar bipartite graph with E edges and $V \geq 3$ vertices. Prove that $E \leq 2V - 4$.

6. (*a*) For which n is K_n planar ?

(*b*) For which m and n is $K_{m, n}$ planar ?

7. Show that $K_{2, 2}$ is homeomorphic to K_3.

8. Suppose a graph G_1 with V_1 vertices and E_1 edges is homeomorphic to a graph G_2 with V_2 vertices and E_2 edges prove that $E_2 - V_2 = E_1 - V_1$.

9. Show that any graph homeomorphic to K_5 or $K_{3, 3}$ is obtainable from K_5 or $K_{3, 3}$ respectively, by addition of vertices to edges.

10. (*a*) Let G be a connected graph with V_1 vertices and E_1 edges and let H be a subgraph with V_2 vertices and E_2 edges. Show that $E_2 - V_2 \leq E_1 - V_1$.

(*b*) Let G be a connected graph with V vertices ; E edges, and $E \leq V + 2$. Show that G is planar.

11. Let G be a graph and let H be obtained from G by adjoining a new vertex of degree 1 to some vertex of G. Is it possible for G and H to be homeomorphic ? Explain.

12. (*a*) Show that any planar graph all of whose vertices have degree at least 5 must have at least 12 vertices.

(*b*) Find a planar graph each of whose vertices has degree at least 5.

13. (*a*) Prove that if G is a planar graph with *n* connected components, each components having at least three vertices then $E \leq 3V - 6n$.

(*b*) Prove that if G is a planar graph with *n* connected components, then it is always true that $E \leq 3V - 3n$.

14. (*a*) Prove that every planar graph with $V \geq 2$ vertices has at least two vertices of degree $d \leq 5$.

(*b*) Prove that every planar graph with $V \geq 3$ vertices has at least three vertices of degree $d \leq 5$.

(*c*) Prove that every planar graph with $V \geq 4$ vertices has at least four vertices of degree of $d \leq 5$.

15. (*a*) A connected planar graph G has 20 vertices. Prove that G has at most 54 edges.

(*b*) A connected planar graph G has 20 vertices, seven of which have degree 1. Prove that G has at most 40 edges.

16. Suppose G is a connected planar graph in which every vertex has degree at least 3. Prove that at least two regions of G have at most five edges on their boundaries.

17. Draw a graph corresponding to the map shown at the right and find a coloring which requires the least number of colors. What is the chromatic number of the graph ?

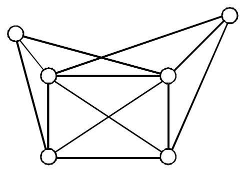
18. (*a*) What is $\chi(K_{14})$? What is $\chi(K_5, 14)$? Why ?

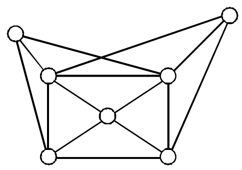
(*b*) Let G_1 and G_2 by cycles with 38 and 107 edges, respectively. What is $\chi(G_1)$? What is $\chi(G_2)$? Explain.

19. Let $n \geq 4$ be a natural number. Let G be the graph which consists of the union of K_{n-3} and a 5-cycle C together with all possible edges between the vertices of these graphs. Show that $\chi(G) = n$, yet G does not have K_n as a subgraph.

20. Find a formula for $V - E + R$ which applies to planar graphs which are not necessarily connected.

21. Find its chromatic number and explain why this piece of information is consistent with the four color problem.





22. Prove that every subgraph of a planar graph is planar.

23. Prove that :

(*i*) K_5 is the non planar graph with the smallest number of vertices.

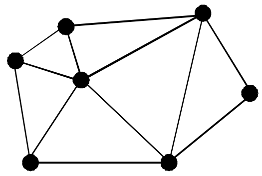
(*ii*) $K_{3,3}$ is the non planar graph with the smallest number of edges.

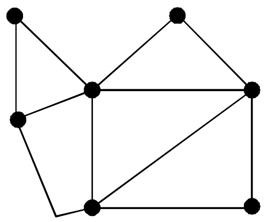
24. Show that every graph with four or fewer vertices is planar.

25. Show that the graphs $K_{1, S}$ for $S \geq 1$ and $K_{2, S}$ for $S \geq 2$ are planar.

26. Let G be a simple connected graph with at least 11 vertices. Prove that either G or its complement $\overline{G}$ must be non planar.

27. Verify the Euler's formula for the graphs shown below :





28. Verify the Euler's formula for the graphs W_8, $K_{1, 5}$ and $K_{2, 7}$.

29. Prove that every simple connected planar graph with $n \geq 4$ vertices has at least four vertices of degree five or less.

30. Let G be a connected planar graph with more than two vertices. If G has exactly n_K vertices of degree K and $\Delta(G) = P$, show that

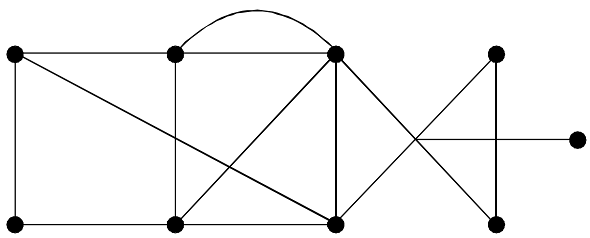
$$5n_1 + 4n_2 + 3n_3 + 2n_4 + n_5 \geq n_7 + 2n_8 + \ldots\ldots + (P - 6)\, n_P + 12.$$

31. Prove that the sum of the degree of the regions of a planar graph is equal to twice the number of edges in the graph.

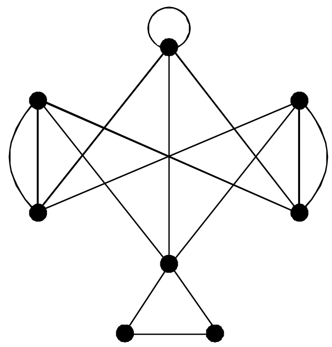
32. Show that a simple planar connected graph with less than 30 edges must have a vertex of degree ≤ 4.

33. What is the minimum number of vertices necessary for a simple connected graph with 7 edges to be planar ?

34. By using the method of elementary reduction, show that the following graph is planar.



35. By the method of elementary reduction, show that the following graph is non planar.



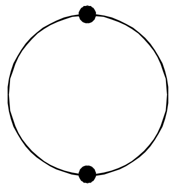
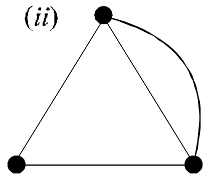
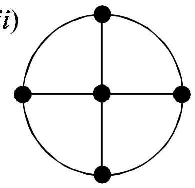
36. Prove that a planar graph G is isomorphic to G** if and only if G is connected.

37. Let G be a planar connected graph. Prove that G is bipartite if and only if G* is an Euler graph.

38. Prove that a self loop free planar graph G is 2-connected if and only if G* is 2-connected.

39. Prove that 5-connected planar graph has at least 12 vertices.

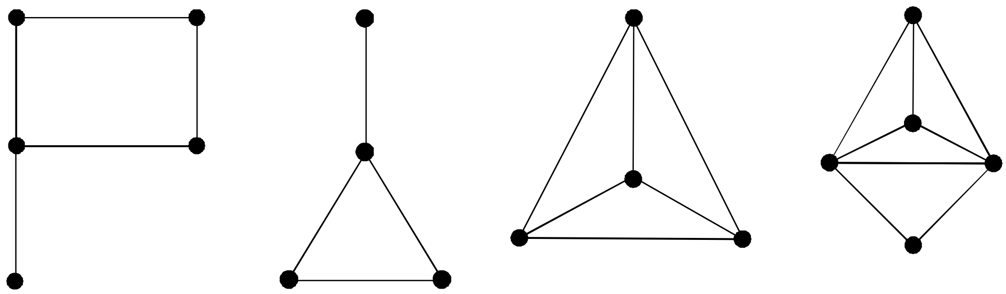
40. Show that the following graphs are self dual.

(*i*)  (*ii*)  (*iii*) 

41. Show that a simple graph with n vertices and more than $\left[\dfrac{n^2}{4}\right]$ edges cannot be 2-chromatic.

42. Prove or disprove that in a graph of order n and size m, $\chi(G) \le 1 + \dfrac{2m}{n}$.

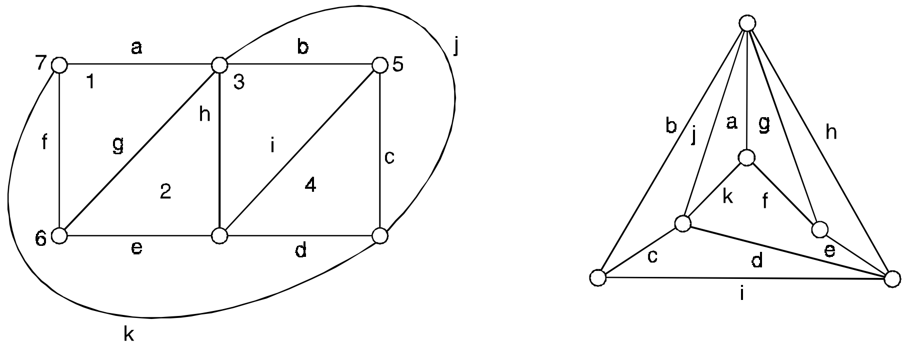
43. Find the chromatic numbers of the following



(*i*) (*ii*) (*iii*) (*iv*)

Answers 2.1

1. (*a*) We draw the graph quickly as a planar and then, after some thinking, as a planar graph with straight edges.



(*b*) There are seven regions, numbered 1, 2, ... 7, with boundaries *afg, ghe, hbi, icd, bjc, fedk*, and *ajk*, respectively.

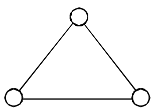
(*c*) E = 11, V = 6, R = 7, N = 22, so V – E + R = 6 – 11 + 7 = 2 ; N = 22 ≤ 22 = 2E and E = 11 ≤ 12 ≤ 3V – 6.

2.

Solid	*V*	*E*	*F*	*V – E + F*
tetrahedron	4	6	4	2
cube	8	12	6	2
octahedron	6	12	8	2
dodecahedron	201	30	12	2
icosahedron	12	30	20	2

3. We know that E ≤ 3V – 6, substituting E = V + R – 2, we obtain V + R – 2 ≤ 3V – 6 or R ≤ 2V – 4, as required.

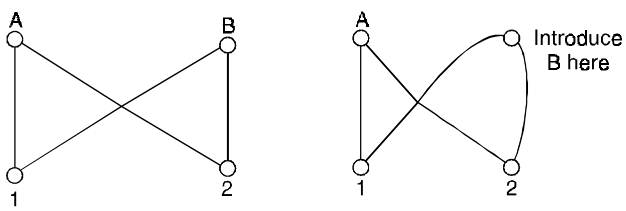
4. (*a*)



E = 3 = 3(3) – 6 = 3V – 6.

6. (*a*) K_n is planar if and only if $n \le 4$.

7. (*a*)



10. (*a*) Assume the result is not true. Then there is some counter example G and subgraph H ; that is, for these graphs $E_1 - V_1 < E_2 - V_2$. (In particular, $V_1 \neq V_2$), choose H such that $V_1 - V_2$ is as small as possible. Since G is connected, we can find a vertex *v* which is in G, but not H which is joined to some vertex in H. Let K be that subgraph of G consisting of H, *v* and all edges joining *v* to vertices in H. Letting v_3 and E_3 denote the number of vertices and edges, respectively, in K, we have $V_3 = V_2 + 1$, while $E_3 \geq E_2 + 1$. Hence $E_2 - V_2 \leq E_3 - V_3$ and so $E_1 - V_1 < E_3 - V_3$. Thus, G and its subgraph K provide another counter example, but this contradicts the minimality of $V_1 - V_2$ since $V_1 - V_3 < V_1 - V_2$.

11. Yes. An example is shown to the right. The graphs are homeomorphic since the one on the right is obtainable from the other by adding a vertex of degree 2.

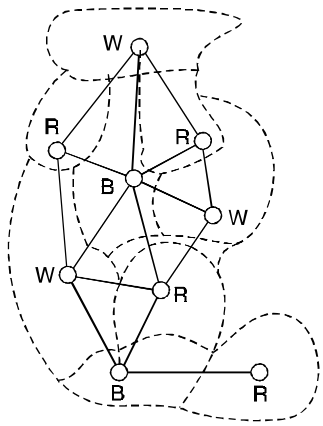
13. (*a*) Let $G_1, G_2, ..., G_n$ be the connected components of G. Since G_i has atleast three vertices, we have $E_{G_i} \leq 3\,V_{G_i} - 6$. Hence, $\Sigma\,E_{G_i} \leq 3\,\Sigma\,V_{G_i} - 6n$, so $E \leq 3V - 6n$ is required.

14. (*a*) We may assume that G is connected. Say there is only one vertex of degree atmost 5. Then
$\Sigma \deg v_i \geq 6(V - 1) = 6V - 6$, contradicting
$\Sigma \deg v_i = 2E \leq 6V - 12$.

15. (*a*) $E \leq 3V - 6$, so $E \leq 3(20) - 6 = 54$.

16. Say at most one region has atmost five edges on its boundary. Then, $N \geq 6(R - 1)$. But $N \leq 2E$, so $2E \geq 6R - 6$, $3R \leq E + 3$. Sinve $V - E + R = 2$, $6 = 3V - 3E + 3R \leq 3V - 2E + 3$ that is, $2E \leq 3V - 3$. But $2E = \Sigma\deg v_i \geq 3V$ by assumption, and this is a contradiction.

17. We show the graph superimposed over the given map. Since this graph contains triangles, atleast three colours are necessary. A 3-colouring is shown, so the chromatic number is 3.



18. (*a*) For any *n*, $\chi(K_n) = n$ and for any *m, n,* $\chi(K_{m,n}) = 2$. Thus, $\chi(K_{14}) = 14$ and $\chi(K_{5,14}) = 2$.

20. Letting *x* denote the number of connected components of G, we have $V - E + R = 1 + x$.

For each component C, $V_C - E_C + R_C = 2$. Adding, we get

$\Sigma V_C - \Sigma E_C + \Sigma R_C = 2x$. We have $\Sigma V_C = V$ and $\Sigma E_C = E$, but $\Sigma R_C = R + (x - 1)$ since the exterior region is common to all components.

Thus, $V - E + R + x - 1 = 2x$, $V - E' + R' = x + 1$.

42. (*i*) 2, (*ii*) 3, (*iii*) 4 (*iv*) 4.

CHAPTER

3 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** $\mathbf{C_n H_{2n+2}}$ with a given number of n of carbon atoms as

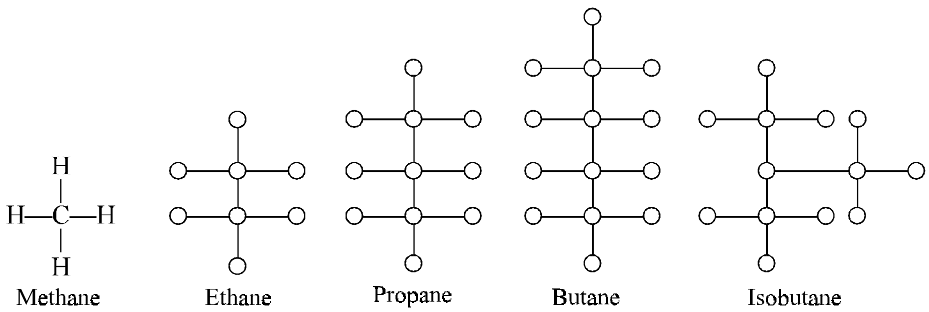


Fig. 3.1.

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 :

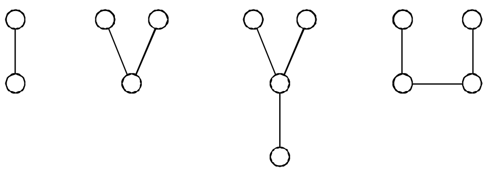


Fig. 3.2.

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

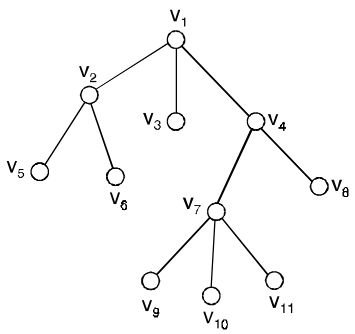


Fig. 3.3. A rooted tree.

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 :

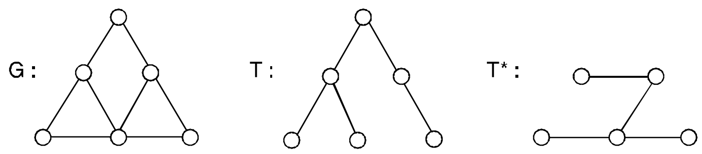


Fig. 3.4.

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

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

(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.

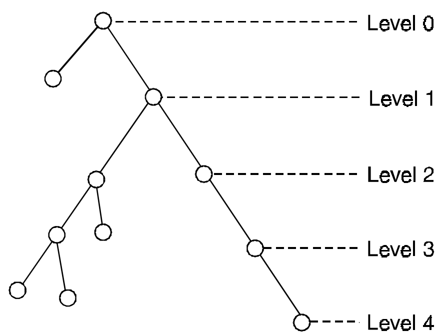


Fig. 3.5. 13-vertices, 4-level binary tree.

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

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,

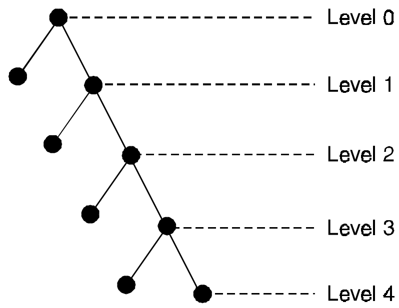


Fig. 3.6.

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,

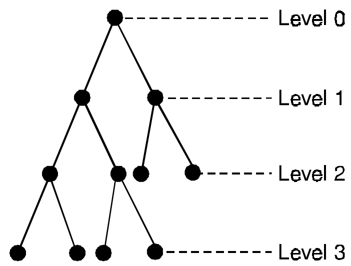


Fig. 3.7.

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,

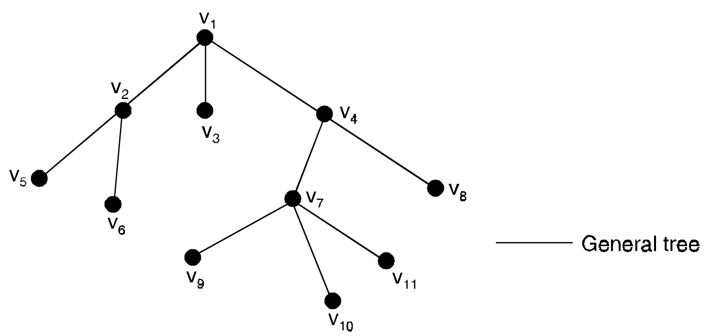


Fig. 3.8.

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.

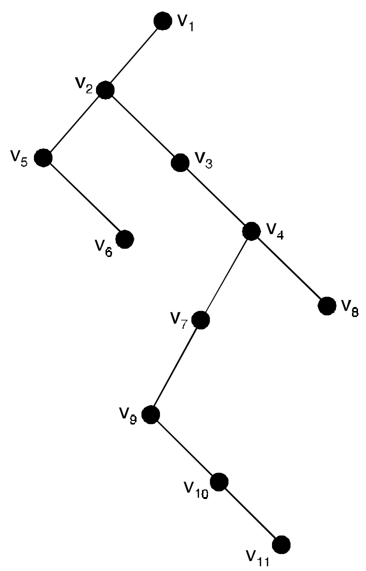


Fig. 3.9.

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.

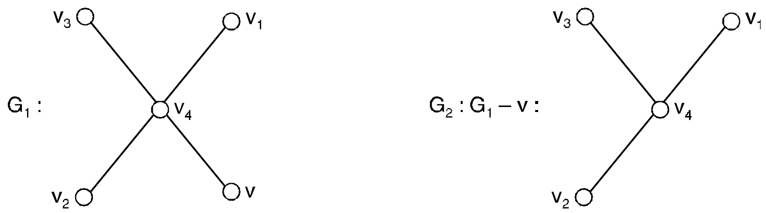


Fig. 3.10.

$$| V(G_1) | = p + 1 = (| E(G_2) | + 1) + 1$$
$$= | E(G_1) | + 1$$
$$\therefore \quad | V(G_1) | = | E(G_1) | + 1.$$

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 $\pi : v_0, v_1, ..., v_k w$ 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 $\pi_1 : v_0 v_1 ...... v_k w_1 w$ and $\pi_2 : v_0 v_1 ...... v_k w_2 w$ 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).

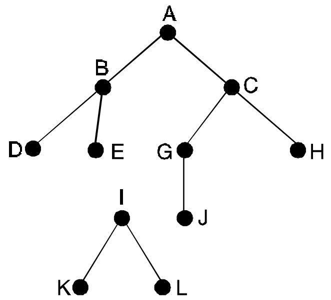


Fig. 3.11

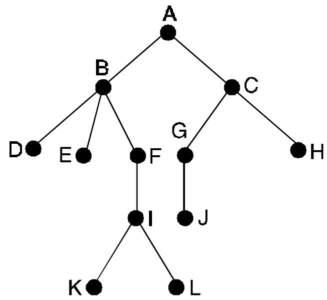


Fig. 3.12

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.

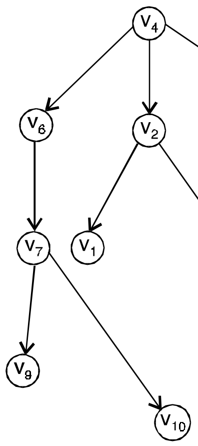


Fig. 3.13

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, ... 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).*

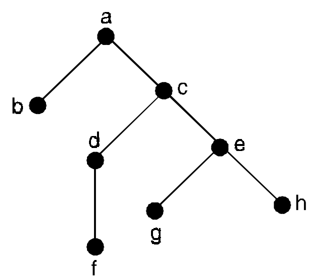


Fig. 3.14.

(*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.

(*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 $m \cdot m^{h-1} = m^h$.

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

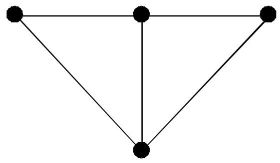


Fig. 3.15.

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).

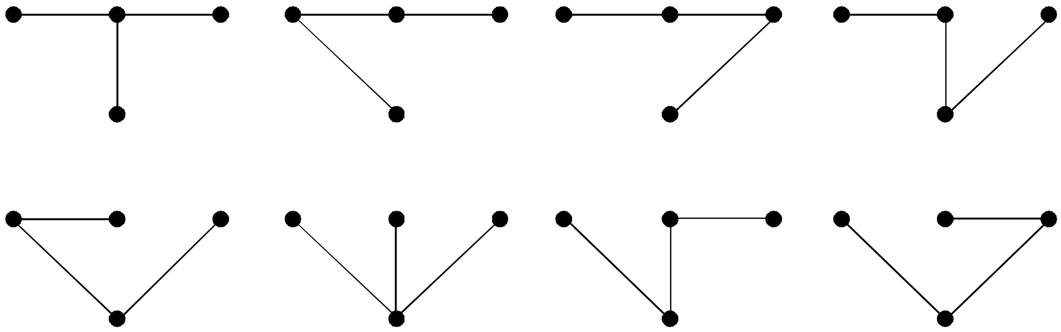


Fig. 3.16.

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

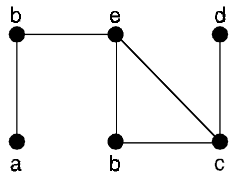


Fig. 3.17.

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.

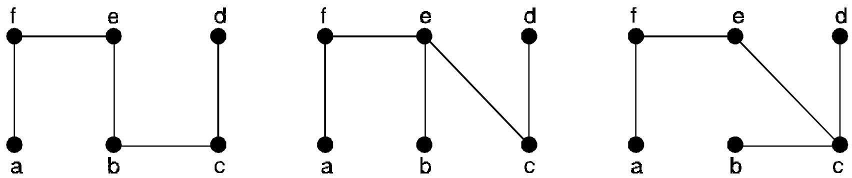


Fig. 3.18.

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.

Theorem 3.14. *There is one and only path between every pair of vertices in a tree.*

(OR)

A graph G is a tree if and only if every two distinct vertices of G are joined by a unique path of G.

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

Now suppose that between two vertices a and b of T there are two distinct paths.

The union of these two paths will contain a cycle, and T cannot be a tree.

Conversely, suppose in a graph G there is one and only one path between every pair of vertices, then G is a tree.

If there exists a path between every pair of vertices, then G is connected.

A cycle in a graph implies that there is atleast one pair of vertices a and b such that there are two distinct paths between a and b.

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

Therefore, G is a tree.

Theorem 3.15. *Every non trivial tree contains atleast two end vertices.*

Proof. Suppose that T is a tree with p-vertices and q-edges and let $d_1, d_2, \ldots\ldots d_p$ denotes the degrees of its vertices, ordered so that $d_1 \le d_2 \le \ldots\ldots \le d_p$.

Since T is connected and non trivial, $d_i \ge 1$ for each $i(1 \le i \le p)$.

If T does not contain two end vertices, then $d_i \ge 1$ and $d_i \ge 2$ for $2 \le i \le p$,

So $$\sum_{i=1}^{p} d_i \ge 1 + 2\,(p-1) = 2p - 1 \qquad \text{...(1)}$$

However from the results *i.e.*, $\sum_{i=1}^{p} \deg v_i = 2q$ and a tree with p-vertices has $p - 1$ edges.

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

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.

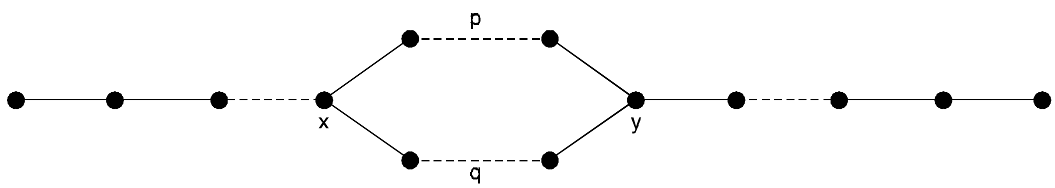


Fig. 3.19.

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 on 6 vertices.*

Solution.

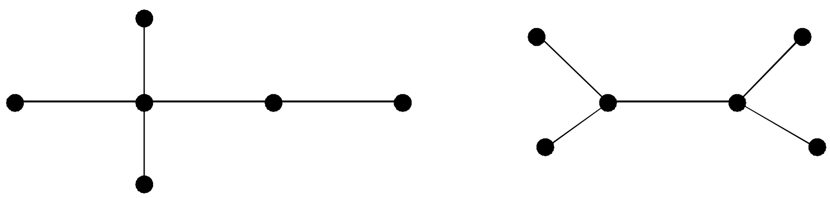


Fig. 3.20.

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.

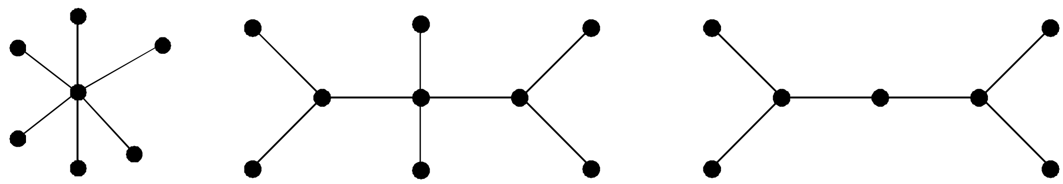


Fig. 3.21.

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

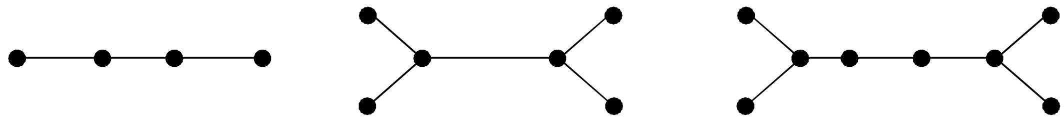


Fig. 3.22.

Problem 3.11. *Count the number of vertices of degree three in a binary tree on n vertices having k number of pendant vertices.*

Solution. Since the binary tree contains k number of pendant vertices and one vertex of degree two, we have total number of remaining vertices which are of degree three is $n - k - 1$.

Problem 3.12. *Let T be a tree with 50 edges. The removal of certain edge from T yields two disjoint trees T_1 and T_2. Given that the number of vertices in T_1 equals the number of edges in T_2, determine the number of vertices and the number of edges in T_1 and T_2.*

Solution. We have removal of an edge from a graph will not remove any vertex from the graph.

Thus $| V(T_1) | + | V(T_2) | = | V(T) |$

Since T_1 and T_2 are trees and number of vertices of T_1 is equal to the number of edges in T_2, we get

$$\begin{aligned} | V(T) | &= | V(T_1) | + | V(T_2) | \\ &= (| V(T_2) | - 1) + | V(T_2) | \\ &= 2 | V(T_2) | - 1 \end{aligned}$$

but $| V(T) | = | E(T) + 1 | = 50 + 1 = 51$

Hence $2 | V(T_2) | - 1 = 51$

$\Rightarrow \quad | V(T_2) | = 26$ and $| V(T_1) | = 25$

Therefore, there are 26 vertices and hence 25 edges in T_2 and there are 25 vertices hence 24 edges in T_1.

Thus $50 - (25 + 24) = 1$ edge is removed from the tree T.

Problem 3.13. *What is the maximum number of end vertices a tree on n vertices may have ?*

Solution. The graph $K_{1, n}$ contains maximum number of end vertices.

Thus a tree on n vertices may contain a maximum of $n - 1$ end vertices.

Problem 3.14. *Prove that a pendant edge in a connected graph G is contained in every spanning tree of G.*

Solution. By a pendant edge, we mean an edge whose one end vertex is a pendant vertex.

Let e be a pendant edge of a connected graph G and let v be the corresponding pendant vertex.

Then e is the only edge that is incident on v.

Suppose there is a spanning tree of T for which e is not a branch.

Then, T cannot contain the vertex v.

This is not possible, because T must contain every vertex of G.

Hence there is no spanning tree of G for which e is not a branch.

Problem 3.15. *Show that a Hamiltonian path is a spanning tree.*

Solution. Recall that a Hamiltonian path P in a connected graph G, if there is a path which contains every vertex of G and that if G has n vertices then P has $n - 1$ edges.

Thus, P is a connected subgraph of G with n vertices and $n - 1$ edges.

Therefore, P is a tree. Since P contains all vertices of G, it is a spanning tree of G.

Problem 3.16. *Prove that the number of branches of a spanning tree T of a connected graph G is equal to the rank of G and the number of the corresponding chords is equal to the nullity of G.*

Solution. Let n be the number of vertices and m be the number of edges in a connected graph G. Then

Rank of G = $\rho(G) = n - 1$

= no. of branches of a spanning tree T of G.

Nullity of G = $\mu(G) = m - (n - 1)$

= no. of chords relative to T.

Problem 3.17. *Prove that every circuit in a graph G must have atleast one edge in common with a chord set.*

Solution. Recall that a chord set is the complement of a spanning tree.

If there is a circuit that has no common edge with this set, the circuit must be containined in a spanning tree.

This is impossible, because a tree does not contain a circuit.

Problem 3.18. *Let G be a graph with k components, where each component is a tree. If n is the number of vertices and m is the number of edges in G, prove that n = m + k.*

Solution. Let $H_1, H_2, \ldots\ldots H_k$ be the components of G.

Since each of these is a tree, if n_i is the number of vertices in H_i and m_i is the number of edges in H_i

We have $\quad m_i = n_i - 1, \quad i = 1, 2, \ldots\ldots k$

this gives $\quad m_1 + m_2 + \ldots\ldots + m_k = (n_1 - 1) + (n_2 - 1) + \ldots\ldots (n_k - 1)$

$$= n_1 + n_2 + \ldots\ldots + n_k - k$$

But $\quad m_1 + m_2 + \ldots\ldots + m_k = m \quad$ and

$$n_1 + n_2 + \ldots\ldots + n_k = n$$

Therefore $\quad m = n - k$

$\Rightarrow \quad n = m + k.$

Problem 3.19. *Show that, in a tree ; if the degree of every non-pendant vertex is 3, the number of vertices in the tree is even.*

Solution. Let n be the number of vertices in a tree T.

Let k be the number of pendant vertices.

Then, if each non-pendant vertex is of degree 3, the sum of the degrees of vertices is $k + 3(n - k)$.

This must be equal to $2(n - 1)$

Thus, $k + 3(n - k) = 2(n - 1)$

$\Rightarrow \quad n = 2(k - 1)$

Therefore, n is even.

Problem 3.20. *Suppose that a tree T has N_1 vertices of degree 1, N_2 vertices of degree 2, N_3 vertices of degree 3, ... N_k vertices of degree k. Prove that*

$$N_1 = 2 + N_3 + 2N_4 + 3N_5 + ... + (K - 2) N_k.$$

Solution. Note that a tree T,

The total number of vertices $= N_1 + N_2 + ... + N_k$

Sum of the degrees of vertices $= N_1 + 2N_2 + 3N_3 + ... kN_k$

Therefore, the total number of edges in T is

$$N_1 + N_2 + ... + N_k - 1, \text{ and}$$

the handshaking property, gives

$$N_1 + 2N_2 + 3N_3 + 4N_4 + 5N_5 + ... kN_k$$
$$= 2(N_1 + N_2 + ... + N_k - 1)$$

Rearranging terms, which gives

$$N_3 + 2N_4 + 3N_5 + ... + (k - 2) N_k = N_1 - 2$$

$\Rightarrow \quad N_1 = 2 + N_3 + 2N_4 + 3N_5 + ... + (k - 2) N_k.$

Problem 3.21. *Show that if a tree has exactly two pendant vertices, the degree of every other vertex is two.*

Solution. Let n be the number of vertices in a tree T.

Suppose, it has exactly two pendant vertices, and let $d_1, d_2, ... d_{n-2}$ be the degrees of the other vertices.

Then, since T has exactly $n - 1$ edges.

We have $\quad 1 + 1 + d_1 + d_2 + ... + d_{n-2} = 2(n - 1)$

$\Rightarrow \quad d_1 + d_2 + ... + d_{n-2} = 2n - 4 = 2(n - 2)$

The left hand side of the above condition has $n - 2$ terms d's, and none of these is one or zero.

Therefore, this condition holds only if each of the $d_i s$ is equal to two.

Problem 3.22. *Show that the complete graph K_n is not a tree, when $n > 2$.*

Solution. If v_1, v_2, v_3 are any three vertices of K_n, $n > 2$ then the closed walk $v_1v_2v_3v_1$ is a circuit in K_n.

Since K_n has a circuit, it cannot be a tree.

Problem 3.23. *Suppose that a tree T has two vertices of degree 2, four vertices of degree 3 and three vertices of degree 4. Find the number of pendant vertices in T.*

Solution. Let N be the number of pendant vertices in T.

It is given that T has two vertices of degree 2, four vertices of degree 3 and three vertices of degree 4.

Therefore, the total number of vertices

$$= N + 2 + 4 + 3$$
$$= N + 9.$$

Sum of the degrees of vertices $= N + (2 \times 2) + (4 \times 3) + (3 \times 4)$

$$= N + 28.$$

Since T has N + 9 vertices, it has N + 9 – 1 = N + 8 edges.

Therefore, by handshaking property, we have

$$N + 28 = 2(N + 8)$$
$$\Rightarrow \quad N = 12$$

Thus, the given tree has 12 pendant vertices.

Problem 3.24. *Show that the complete bipartite graph $K_{r,s}$ is not a tree if $r \geq 2$.*

Solution. Let v_1 and v_2 be any two vertices in the first partition and v_1', v_2' be any two vertices in the second partition of $K_{r,s}$ $s \geq r > 1$.

Then the closed walk $v_1v_1'v_2v_2'v_1$ is a circuit in $K_{r,s}$.

Since $K_{r,s}$ has a circuit, it cannot be a tree.

Problem 3.25. *Prove that, in a tree with two or more vertices, there are atleast two leaves (pendant vertices).*

Solution. Consider a tree T with n vertices, $n \geq 2$. Then, it has $n - 1$ edges.

Therefore, the sum of the degrees of the n vertices must be equal to $2(n - 1)$.

Thus, if $d_1, d_2, ... d_n$ are the degrees of vertices.

We have $\quad d_1 + d_2 + ... + d_n = 2(n - 1) = 2n - 2.$

If each of $d_1, d_2, ... d_n$ is ≥ 2, then their sum must be at least $2n$.

Since this is not true, atleast one of the d's is less than 2.

Thus, there is a d which is equal to 1.

Without loss of generality, let us take this to be d_1. Then

$$d_2 + d_3 + ... + d_n = (2n - 2) - 1 = 2n - 3.$$

This is possible only if atleast one of $d_2, d_3 ...... d_n$ is equal to 1.

So, there is atleast one more d which is equal to 1.

Thus, there are atleast two vertices with degree 1.

Problem 3.26. *Prove that a graph with n vertices, n – 1 edges, and no circuits is connected.*

Solution. Consider a graph G which has n vertices, $n - 1$ edges and no circuits.

Suppose G is not connected.

Let the components of G be H_i, $i = 1, 2, ... k$.

If H_i has n_i vertices, we have

$$n_1 + n_2 + ... + n_k = n.$$

Since G has no circuits, $H_i s$ is also do not have circuits.

Further, they are all connected graphs.

Therefore, they are trees.

Consequently, each H_i must have $n_i - 1$ edges.

Therefore, the total number of edges in these $H_i s$ is $(n_1 - 1) + (n_2 - 1) + \ldots\ldots (n_k - 1) = n - k$.

This must be equal to the total number of edges in G, that is $n - k = n - 1$.

This is not possible, since $k > 1$.

Therefore, G must be connected.

Problem 3.27. *Construct three distinct binary trees on 11 vertices.*

Solution. (*i*) (*ii*) (*iii*)

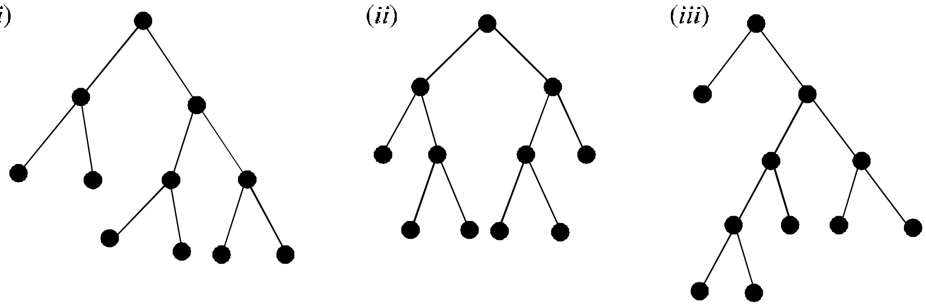


Fig. 3.23.

Problem 3.28. *What is the minimum possible height of a binary tree on 2n – 1 (n ≥ 1) vertices ?*

Solution. Let k be the minimum height of a binary tree on $2n - 1$ vertices.

For minimum height we have to keep maximum number of vertices in the previous level before placing any vertex in the next level.

Thus, k should satisfy the inequality

$$2n - 1 \le 2^0 + 2^1 + 2^2 + \ldots\ldots + 2^k$$

$$= \frac{1(1-2^{k+1})}{1-2}$$

Since right hand side is a G.P. series with first term is 1 and common ratio having $k + 1$ terms.

i.e., $2n - 1 \le 2^{k+1} - 1 \quad \Rightarrow \quad 2n \le 2^{k+1}$

$$\Rightarrow \quad n \le 2^k.$$

Now taking natural log on both sides we get

$$\log_2 n \le k \quad \Rightarrow \quad k \ge \log_2 n.$$

Since k is an integer, this implies that the minimum value of $k = [\log_2 n]$.

Problem 3.29. *What is the maximum possible number of vertices in any k-level tree ?*

Solution. The level of a root is zero and it is the only one vertex at level zero.

There are two vertices that are adjacent to the root, at which are at levels one.

From these vertices we can find maximum four vertices at level 2 so on ...... to get a minimum heighten tree we have to keep the vertex at higher level only after filling all the vertices in its lower level.

Trees maximum number of vertices possible for such a k-level tree is therefore

$$n \le 2^0 + 2^1 + 2^2 + \ldots\ldots 2^k = \frac{1(1-2^{k+1})}{1-2} = 2^{k+1} - 1.$$

Problem 3.30. *What is the maximum possible level (height) of a binary tree on 2n + 1 (n ≥ 0) vertices.*

Solution. Let k be the height of a binary tree on $2n + 1$ vertices.

To get a vertex in maximum level we must keep exactly two (minimum) vertices in each level except the root vertex.

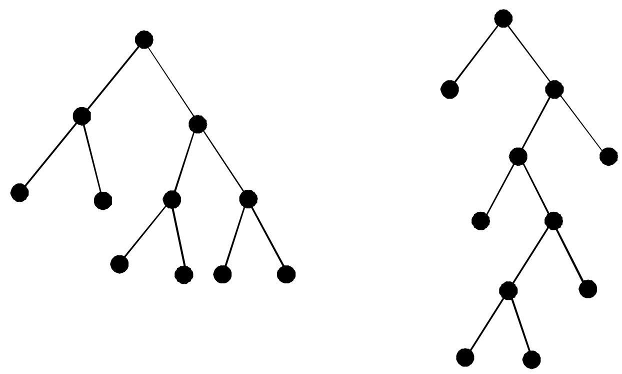
That is out of $2n + 1$ vertices one is a root and the remaining $2n$ vertices can keep in exactly n levels.

Thus the maximum height of a tree is n.

Hence maximum possible value of k is n.

Problem 3.31. *Sketch two different binary trees on 11 vertices with one having maximum height and the other with minimum height.*

Solution. Required binary trees on 11 vertices are



(*i*) **With minimum height 3** (*ii*) **With maximum height 5.**

Fig. 3.24.

Problem 3.32. *Show that the number of vertices in a binary tree is always odd.*

Solution. Consider a binary tree on n vertices. Since it contains exactly one vertex of degree two and other vertices are of degree one or three, it follows that there are $n - 1$ odd degree vertices in the graph.

But if the number of odd degree vertices of a graph is even, it follows that $n - 1$ is even and hence n is odd.

Problem 3.33. *In any binary tree T on n vertices, show that the number of pendant vertices (edges) is equal to* $\dfrac{(n+1)}{2}$.

Solution. Let the number of pendant edges in a binary tree on n vertices be k.

Then we have there are $n - k - 1$ vertices of degree three, one vertex of degree two, k vertices of degree one and $n - 1$ edges.

Therefore, sum of degrees of vertices = 2 × number of edges.

$$(n - k - 1) \times 3 + 2 + k \times 1 = 2(n - 1)$$

$$\Rightarrow \quad 3n - 3k - 3 + 2 + k = 2n - 2$$

$$\Rightarrow \quad 2k = 3n - 2n + 1 = n + 1$$

$$\Rightarrow \quad k = \frac{(n+1)}{2}.$$

Problem 3.34. *Draw a tree with 6 vertices, exactly 3 of which have degree 1.*

Solution. A tree with 6 vertices which contains 3 pendant vertices is given in Figure (3.25).

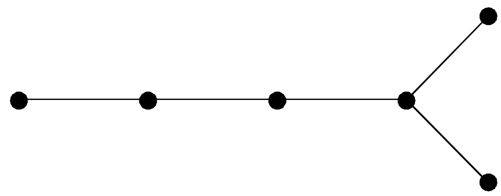


Fig. 3.25.

Problem 3.35. *Which trees are complete bipartite graphs ?*

Solution. Suppose T is a tree which is a complete bipartite graph.

Let T = $K_{m,\,n}$ then the number of vertices in T is ($m + n$).

Hence the tree T contains ($m + n - 1$) number of edges.

But the graph $K_{m,\,n}$ has (m, n) number of edges.

Therefore $\quad m + n - 1 = mn$

$$\Rightarrow \quad mn - m - n + 1 = 0$$

$$\Rightarrow \quad m(n - 1) - 1(n - 1) = 0$$

$$\Rightarrow \quad (m - 1)(n - 1) = 0$$

$$\Rightarrow \quad m = 1 \text{ or } n = 1$$

This means T is either $K_{1,\,n}$ or $K_{m,\,1}$ that is T is a star.

Problem 3.36. *Draw all non-isomorphic trees with 6 vertices.*

Solution. All non isomorphic trees with 6 vertices are shown below :

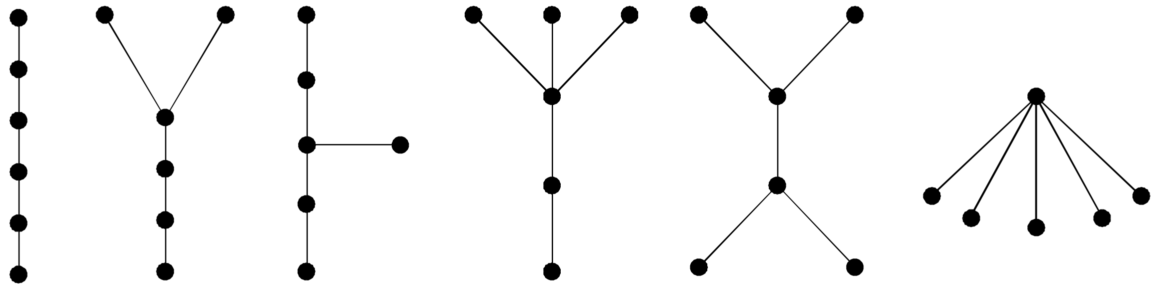


Fig. 3.26.

Problem 3.37. *Is it possible to draw a tree with five vertices having degrees 1, 1, 2, 2, 4.*

Solution. Since the tree has 5 vertices hence it has 4 edges.

Now given the vertices of tree are having degrees

1, 1, 2, 2, 4.

i.e., the sum of the degrees of the tree = 10

By handshaking lemma, $2q = \sum_{i=1}^{5} d(v_i)$

Where q is the number of edges in the graph

$2q = 10 \quad \Rightarrow \quad q = 5$

Which is contradiction to the statement that the tree has 4 edges with 5 vertices.

Hence the tree with given degrees of vertices does not exist.

3.5. COUNTING TREES

The subject of graph enumeration is concerned with the problem of finding out how many non-isomorphic graphs possess a given property. The subject was initiated in the 1850's by Arthur Cayley, who later applied it to the problem of enumerating alkanes $C_n H_{2n+2}$ with a given number of carbon atoms. This problem is that of counting the number of trees in which the degree of each vertex is either 4 or 1. Many standard problems of graph enumeration have been solved.

For example, it is possible to calculate the number of graphs, connected graphs, trees and Eulerian graphs with a given number of vertices and edges, corresponding general results for planar graphs and Hamiltonian graphs have, however, not yet been obtained. Most of the known results can be obtained by using a fundamental enumeration theorem due to Polya, a good account of which may be found in Harary and Palmer.

Unfortunately, in almost every case it is impossible to express these results by means of simple formulas.

Consider Fig. (3.27), which shows three ways of labelling a tree with four vertices. Since the second labelled tree is the reverse of the first one, these two labelled trees are the same. On the other hand, neither is isomorphic to the third labelled tree, as you can see by comparing the degrees of vertex 3.

Thus, the reverse of any labelling does not result in a new one, and so the number of ways of labelling this tree is $\frac{(4!)}{2}$ = 12.

Similarly, the number of ways of labelling the tree in Fig. (3.28) is 4, since the central vertex can be labelled in four different ways, and each one determines the labelling.

Thus, the total number of non-isomorphic labelled trees on four vertices is 12 + 4 = 16.

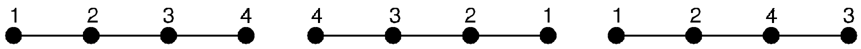


Fig. 3.27.

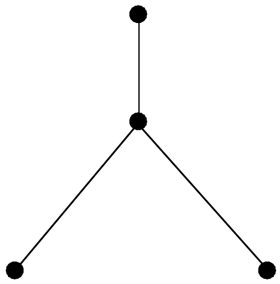


Fig. 3.28.

Theorem 3.19. *Let T be a graph with n vertices. Then the following statements are equivalent :*

(*i*) *T is a tree*

(*ii*) *T contains no cycles, and has n – 1 edges*

(*iii*) *T is connected and has n – 1 edges*

(*iv*) *T is connected and each edge is a bridge*

(*v*) *Any two vertices of T are connected by exactly one path*

(*vi*) *T contains no cycles, but the addition of any new edge creates exactly one cycle.*

Proof. If $n = 1$, all six results are trivial, we therefore assume that $n \geq 2$.

(*i*) ⇒ (*iii*)

Since T contains no cycles, the removal of any edge must disconnect T into two graphs, each of which is a tree.

It follows by induction that the number of edges in each of these two trees is one fewer than the number of vertices. We deduce that the total number of edges of T is $n - 1$.

(*ii*) ⇒ (*iii*)

If T is disconnected, then each component of T is a connected graph with no cycles and hence, by the previous part, the number of vertices in each component exceeds the number of edges by 1.

It follows that the total number of vertices of T exceeds the total number of edges by atleast 2, contradicting the fact that T has $n - 1$ edges.

(iii) $\Rightarrow$ (iv)

The removal of any edge results in a graph with n vertices and $n - 2$ edges, which must be disconnected.

(iv) $\Rightarrow$ (v)

Since T is connected, each pair of vertices is connected by atleast one path.

If a given pair of vertices is connected by two paths, then they enclose a cycle, contradicting the fact that each edge is a bridge.

(v) $\Rightarrow$ (vi)

If T contained a cycle, then any two vertices in the cycle would be connected by atleast two paths, contradicting statement (v).

If an edge e is added to T, then, since the vertices incident with e are already connected in T, a cycle is created.

The fact that only one cycle is obtained.

(vi) $\Rightarrow$ (i)

Suppose that T is disconnected.

If we add to T any edge joining a vertex of one component to a vertex in another, then no cycle is created.

Corollary :

If G is a forest with n vertices and k components, then G has $n - k$ edges.

Theorem 3.20. *If T is any spanning forest of a graph G, then*

(i) *each cutset of G has an edge in common with T*

(ii) *each cycle of G has an edge in common with the complement of T.*

Proof. (i) Let C* be a cutset of G, the removal of which splits a component of G into two subgraphs H and K.

Since T is a spanning forest, T must contain an edge joining a vertex of H to a vertex of K, and this edge is the required edges.

(ii) Let C be a cycle of G having no edge is common with the complement of T.

Then C must be contained in T, which is a contradiction.

3.5.1. Cayley theorem (3.21)

There are n^{n-2} distinct labelled trees on n vertices.

Remark. The following proofs are due to Prüfer and Clarke.

Proof. First proof :

We establish a one-one correspondence between the set of labelled trees of order n and set of sequences $(a_1, a_2, ... a_{n-2})$, where each a_i is an integer satisfying $1 \leq a_i \leq n$.

Since there are precisely n^{n-2} such sequence, the result follows immediately.

We assume that $n \geq 3$, since the result is trivial if $n = 1$ or 2.

In order to establish the required correspondence, we first let T be a labelled tree of order n, and show how the sequence can be determined.

If b_1 is the smallest label assigned to an end-vertex, we let a_1 be the label of the vertex adjacent to the vertex b_1.

We then remove the vertex b_1 and its incident edge, leaving a labelled tree of order $n - 1$.

We next let b_2 be the smallest label assigned to an end-vertex of our new tree, and let a_2 be the label of the vertex adjacent to the vertex b_2.

We then remove the vertex b_2 and its incident edge, as before.

We proceed in this way until there are only two vertices left, and the required sequence is $(a_1, a_2, \ldots\ldots a_{n-2})$.

For example, if T is the labelled tree in Figure (3.29),

then $b_1 = 2, a_1 = 6, b_2 = 3, a_2 = 5, b_3 = 4, a_3 = 6$

$b_4 = 6, a_4 = 5, b_5 = 5, a_5 = 1$

The required sequence is therefore (6, 5, 6, 5, 1)

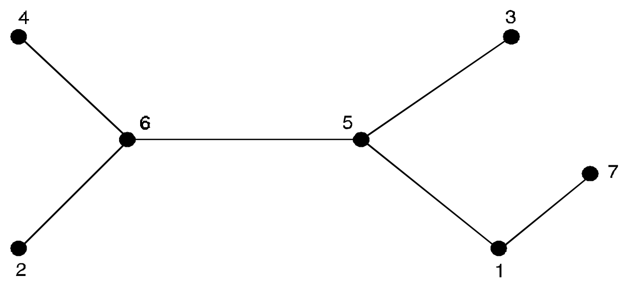


Fig. 3.29.

To obtain the reverse correspondence, we take a sequence $(a_1, \ldots\ldots a_{n-2})$.

Let b_1 be the smallest number that does not appear in it, and join the vertices a_1 and b_1.

We then remove a_1 from the sequence, remove the number b_1 from consideration, and proceed as before.

In this way we build up the tree, edge by edge,

For example, if we start with the sequence (6, 5, 6, 5, 1), then $b_1 = 2, b_2 = 3, b_3 = 4, b_4 = 6, b_5 = 5$, and the corresponding edges are 62, 53, 64, 56, 15.

We conclude by joining the last two vertices not yet crossed out-in this case, 1 and 7.

It is simple to check that if we start with any labelled tree, find the corresponding sequence, and then find the labelled tree corresponding to that sequence, then we obtain the tree we started from.

We have therefore established the required correspondence and the result follows.

Second Proof :

Let $T(n, k)$ be the number of labelled trees on n vertices in which a given vertex v has degree k.

We shall derive an expression for $T(n, k)$, and the result follows on summing from $k = 1$ to $k = n - 1$.

Let A be any labelled tree in which deg $(v) = k - 1$.

The removal from A of any edge wz that is not incident with v leaves two subtrees, one containing v and either w or z (w, say), and the other containing z.

If we now join the vertices v and z, we obtain a labelled tree B in which deg (v) = K see Fig. (3.30).

We call a pair (A, B) of labelled trees of linkage if B can be obtained from A by the above construction.

Our aim is to count the possible linkages (A, B).

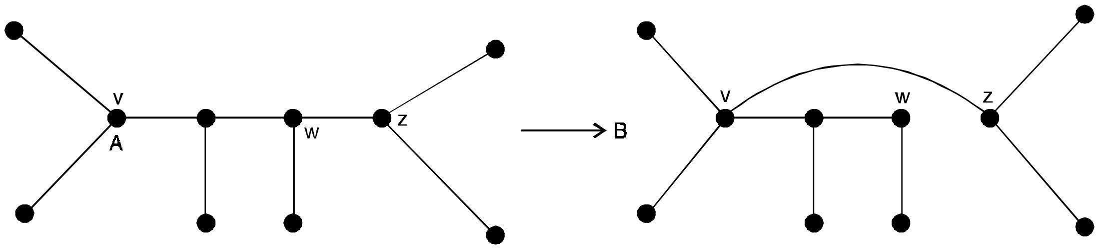


Fig. 3.30.

Since A may be chosen in T(n, k – 1) ways, and since B is uniquely defined by the edge wz which may be chosen in (n – 1) – (k – 1) = (n – k) ways, the total number of linkages (A, B) is (n – k) T(n, k – 1).

On the other hand, let B be a labelled tree in which deg (v) = k, and let T_1, ... T_k be the subtrees obtained from B by removing the vertex v and each edge incident with v.

Then we obtain a labelled tree A with deg (v) = k – 1 by removing from B just one of these edges (vw_i, say, where w_i lies in T_i), and joining w_i to any vertex u of any other subtree T (see Fig. 3.31).

Note that the corresponding pair (A, B) of labelled trees is a linkage, and that all linkages may be obtained in this way.

Since B can be chosen in T(n, k) ways, and the number of ways of joining w_i to vertices in any other T_j is (n – 1) – n_i, where n_i is the number of vertices of T_i, the total number of linkages (A, B) is

$$T(n, k) \{(n - 1 - n_1) + \ldots\ldots + (n - 1 - n_k)\} = (n - 1)(k - 1)\, T(n, k), \text{ since } n_1 + \ldots\ldots + n_k = n - 1$$

We have thus shown that

$$(n - k)\, T(n, k - 1) = (n - 1)(k - 1)\, T(n, k).$$

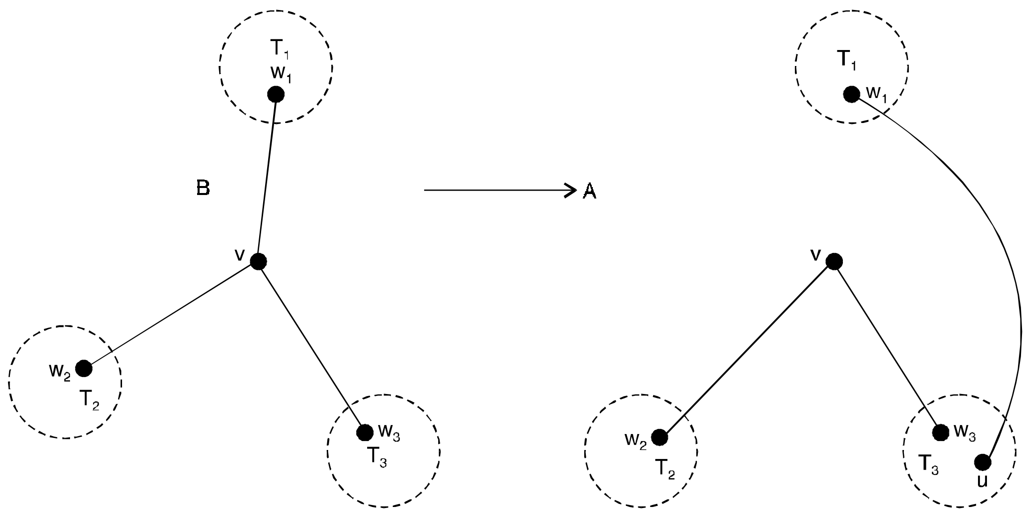


Fig. 3.31.

On iterating this result, and using the obvious fact that T(n, n – 1) = 1, we deduce immediately that

$$T(n, k) = \binom{n-2}{n-1} (n-1)_{n-k-1}$$

On summing over all possible values of k, we deduce that the number T(n) of labelled trees on n vertices is given by

$$T(n) = \sum_{k=1}^{n-1} T(n, k) = \sum_{k=1}^{n-1} \binom{n-2}{k-1}^{n-k-1}$$

$$= \{(n-1)+1\}^{n-2} = n^{n-2}.$$

Corollary :

The number of spanning trees of K_n is n^{n-2}.

Proof. To each labelled tree on n vertices there corresponds a unique spanning tree of K_n.

Conversely, each spanning tree of K_n gives rise to a unique labelled tree on n vertices.

Theorem 3.22. *Prove that the maximum number of vertices in a binary tree of depth d is* $2^d - 1$*, where* $d \geq 1$.

Proof. We shall prove the theorem by induction.

Basis of induction :

The only vertex at depth d = 1 is the root vertex.

Thus the maximum number of vertices on depth

$d = 1$ is $2^1 - 1 = 1$.

Induction hypothesis :

We assume that the theorem is true for depth k,

$d > k \geq 1$

Therefore, the maximum number of vertices on depth k is $2^k - 1$.

Induction step :

By induction hypothesis, the maximum number of vertices on depth k – 1 is $2^{k-1} - 1$.

Since, we know that each vertex in a binary tree has maximum degree 2, therefore, the maximum number of vertices on depth $d = k$ is twice the maximum number of vertices on depth k – 1.

So, at depth k, the maximum number of vertices is $2.2^{k-1} - 1 = 2^k - 1$.

Hence proved.

Problem 3.38. *What are the left and right children of b shown in Fig. 3.32 ? What are the left and right subtrees of a ?*

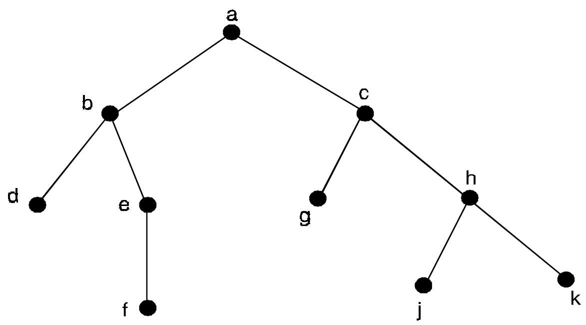
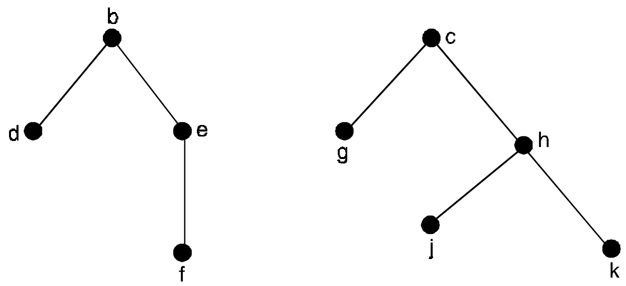


Fig. 3.32.

Solution. The left child of b is d and the right child is e. The left subtree of the vertex a consists of the vertices b, d, e and f and the right subtree of a consists of the vertices c, g, h, j and k whose figures are shown in Fig. 3.33. (a) and (b) respectively.



(a) (b)

Fig. 3.33.

Theorem 3.23. *Prove that the maximum number of vertices on level n of a binary tree is* 2^n*, where* $n \geq 0$*.*

Proof. We prove the theorem by mathematical induction.

Basis of induction :

When $n = 0$, the only vertex is the root.

Thus the maximum number of vertices on level $n = 0$ is $2^0 = 1$.

Induction hypothesis :

We assume that the theorem is true for level K, where $n \geq k \geq 0$.

So the maximum number of vertices on level k is 2^k.

Induction step :

By induction hypothesis, maximum number of vertices on level $k - 1$ is 2^{k-1}.

Since each vertex in binary tree has maximum degree 2, then the maximum number of vertices on level k is twice the maximum number of level $k - 1$.

Hence, the maxmum number of vertices at level k is $= 2.2^{k-1} = 2^k$.

Hence, the theorem is proved.

Theorem 3.24. *If T is full binary tree with i internal vertices, then T has i + 1 terminal vertices and 2i + 1 total vertices.*

Proof. The vertices of T consists of the vertices that are children (of some parent O) and the vertices that are not children (of any parent).

There is one non child-the root. Since there are i internal vertices, each parent having two children, there are $2i$ children.

Thus, there are total $2i + 1$ vertices and the number of terminal vertices is $(2i + 1) – i = i + 1$.

3.6. TREE TRAVERSAL

A traversal of a tree is a process to traverse a tree in a systematic way so that each vertex is visited exactly once. Three commonly used traversals are preorder, postorder and inorder. We describe here these three process that may be used to traverse a binary tree.

3.6.1. Preorder traversal

The preorder traversal of a binary tree is defined recursively as follows

(*i*) Visit the root

(*ii*) Traverse the left subtree in preorder.

(*iii*) Traverse the right subtree in preorder

3.6.2. Postorder traversal

The postorder traversal of a binary tree is defined recursively as follows

(*i*) Traverse the left subtree in postorder

(*ii*) Traverse the right subtree in postorder

(*iii*) Visit the root.

3.6.3. Inorder traversal

The inorder traversal of a binary tree is defined recursively as follows

(*i*) Traverse in inorder the left subtree

(*ii*) Vist the root

(*iii*) Traverse in inorder the right subtree

Given an order of traversal of a tree it is possible to construct a tree.

For example,

Consider the following order :

Inorder = $d\ b\ e\ a\ c$

We can construct the binary trees shown below in Fig. (3.36) using this order of traversal.

3.7. COMPLETE BINARY TREE

If all the leaves of a full binary tree are at level d, then we call a tree as a complete binary tree of depth d. A complete binary tree of depth of 3 is shown in Fig. (3.34).

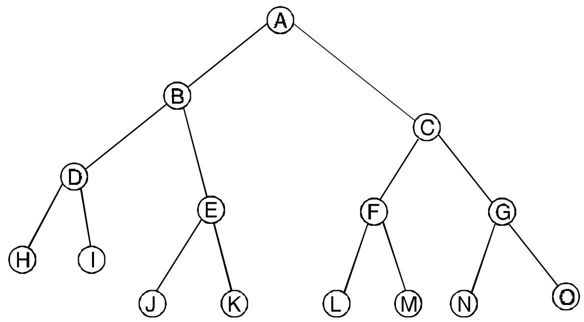


Fig. 3.34. A complete binary tree

3.7.1. Almost complete binary tree

A binary tree of depth *d* is said to be almost complete binary tree if

(*i*) each node in the tree is either at level *d* or *d* – 1.

(*ii*) for any node in the tree with a right descendant at level *d*, all the left descendants of this node are also at level *d*.

Fig. (3.35) shows as almost complete binary tree.

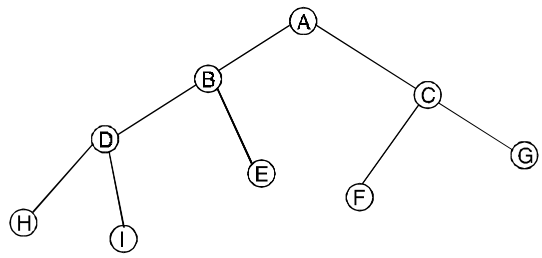


Fig. 3.35. An almost complete binary tree.

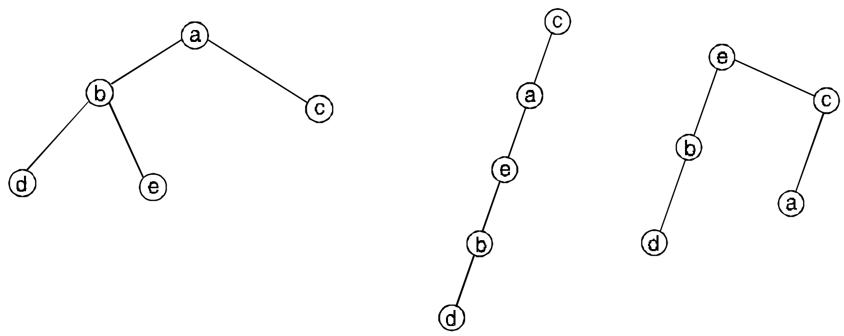


Fig. 3.36. Binary trees constructed using given inorder.

Therefore we can conclude that given only one order of traversal of a tree it is possible to construct a number of binary trees, a unique binary tree is not possible to be constructed.

For construction of a unique binary tree we require two orders in which one has to be inorder, the other can be preorder or postorder.

To draw a unique binary tree when inorder and preorder traversal of the tree is given :

(*i*) The root of T is obtained by choosing the first vertex in its preorder.

(*ii*) The left child of the root vertex is obtained as follows. First use the inroder traversal to find the vertices in the left subtree of the binary tree (all the vertices to the left of this vertex in the inorder traversal are the part of the left subtree).

The left child of the root is obtained by selecting the first vertex in the preorder traversal of the left subtree draw the left child.

(*iii*) Use the inorder traversal to find the vertices in the right subtree of the binary tree (all the vertices to the right of the first vertex are the part of the right subtree).

Then the right child is obtained by selecting the first vertex in the preorder traversal of the right subtree. Draw the right child.

(*iv*) The procedure is repeated recursively until every vertex is not visited in preorder.

To draw a unique binary tree when inorder and postorder traversal of the tree is given :

(*i*) The root of the binary tree is obtained by choosing the last vertex in the postorder traversal.

(*ii*) The right child of the root vertex is obtained as follows. First use the inorder traversal to find the vertices in the right subtree. (all the vertices right to the root vertex in the inorder traversal are the vertices of the right subtree).

The right child of the root is obtained by selecting the last vertex in the postorder traversal. Draw the right child.

(*iii*) Use the inorder traversal to find the vertices in the left subtree of the binary tree. Then the left child is obtained by selecting the last vertex in the postorder traversal of the left subtree. Draw the left child.

(*iv*) The process is repeated recursively until every vertex is not visited in postorder.

3.7.2. Representation of algebraic structure of binary trees

Binary trees are used to represent algebraic expressions, the vertices of the tree are labeled with the numbers, variables or operations that makeup the expression. The leaves of the tree can be labeled with numebrs or variables operations such as addition subtraction, multiplication, division or exponentiation can only be assigned to internal vertices. The operation at each vertex operates on its left and right subtrees from left to right.

3.8. INFIX, PREFIX AND POSTFIX NOTATION OF AN ARITHMETIC EXPRESSION

We know that even for fully parenthesised expression a repeated scanning of the expression is still required in order to evaluate the expression. This phenomenon is due to the fact that operators appear with the operands inside the expression. We can represent expressions in three different ways. They are Infix, Prefix and Postfix forms of an expression.

3.8.1. Infix notation

The notation used in writing the operator between its operands is called infix notation.

The infix form of an algebraic expression is the inorder traversal of the binary tree representing the expression. It gives the original expression with the elements and operations in the same order as they originally occured. To make the infix forms of an expression unambiguous it is necessary to include parentheses in the inorder traversal whenever we encounter an operation.

3.8.2. Prefix notation

The repeated scanning of an infix expression is avoided if it is converted first to an equivalent parenthesis free of polish notaiton. The prefix form of an expression is the preorder traversal of the binary tree representing the given expression. The expression in prefix notation are unambiguous, so that no parentheses are needed in such expression.

3.8.3. Postfix notation

The postfix form of an expression is the postorder traversal of the binary tree representing the given expression. Expressions written in postfix form are said to be in reverse polish notation. Expressions in this notation are unambiguous, so that parentheses are not needed.

Table below gives the equivalent forms of several fully parenthesised expressions. Note that in both the prefix and postfix equivalents of such an infix expression, the variable names are all in the same relative position.

Infix	*Prefix*	*Postfix*
$(x * y) + z)$	$+ xyz$	$xy{*}z +$
$((x + y) * (z + t)$	$* + xy + zt$	$xy + zt + *$
$((x + y * z) - (u/v + w))$	$- + x * yz + uvw$	$xyz * + uv/w + -$

3.8.4. Evaluating prefix and postfix form of an expression

To evaluate an expression in prefix form, proceed as follow move from left to right until we find a string of the form F_{xy}, where F is the symbol for a binary operator and x and y are two operands. Evaluate $x\mathrm{F}y$ and substitute the result for the string F_{xy}. Consider the result as a new operand and continue this procedure until only one number remains. When an expression is in postfix form, it is evaluated in a manner similar to that used for prefix form, except that the operator symbol is after its operands rather than before them.

Problem 3.39. *Represent the expression as a binary tree and write the prefix and postfix forms of the expression*

$$A * B - C \uparrow D + E/F$$

Solution. The binary tree representing the given expression is shown below.

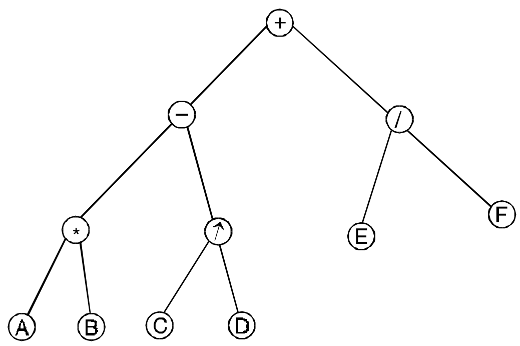


Fig. 3.37.

Prefix : + – * AB ↑ CD/EF

Postfix : AB * CD ↑ – EF/+

Problem 3.40. *What is the value of*

(*a*) *Prefix expression* *x – 84 + 6/42*

(*b*) *Postfix expression* *823 * 2 ↑ 63/+*

Solution. (*a*) The evaluation is carried out in the following sequence of steps

(*i*) *x* – 84 + 6/42

(*ii*) *x*4 + 6/42 since the first string in the F_{xy} is – 84 and 8 – 4 = 4.

(*iii*) *x*4 + 62 replacing /42 by 4/2 = 2

(*iv*) *x*48 replacing + 62 = 8

(*v*) 32 replacing *x*48 by 4*x*8 = 32

(*b*) The evaluation is carried out in the following sequence of steps.

(*i*) 823 * – 2 ↑ 63/+

(*ii*) 86 – 2 ↑ 63/+ replacing 23* by 2 * 3 = 6

(*iii*) 22 ↑ 63/+ replacing 86 – by 8 – 6 = 2

(*iv*) 463/ replacing 2 ↑ 2 by 2^2 = 4

(*v*) 42+ replacing 63/by 6/3 = 2

(*vi*) 6 replacing 42+ by 4 + 2 = 6.

3.9. BINARY SEARCH TREES

A binary search tree is basically a binary tree, and therefore it can be traversed in preorder postorder, and inorder. If we traverse a binary search tree in inorder and print the identifiers contained in the vertices of the tree, we get a sorted list of identifiers in the ascending order.

Binary trees are used extensively in computer science to store elements from an ordered set such as a set of numbers or a set of strings.

Suppose we have a set of strings and numbers. We call them as keys. We are interested in two of the many operations that can performed on this set.

(*i*) ordering (or sorting) the set

(*ii*) searching the ordered set to locate a certain key and, in the event of not finding the key in the set, adding it at the right position so that the ordering of the set is maintained.

A binary search tree is a binary tree T in which data are associated with the vertices. The data are arranged so that, ∈ for each vertex *v* in T, each data item in the left subtree of *v* is less than the data item in *v* and each data item in the right subtree of *v* is greater than the data item in *v*.

Thus, a binary search tree for a set S is a label binary tree in which each vertex *v* is labelled by an element $l(v) \in$ S such that

(*i*) for each vertex *u* in the left subtree of *v*, $l(u) < l(v)$,

(*ii*) for each vertex *u* in the right subtree of *v*, $l(u) > l(v)$,

(*iii*) for each element $a \in$ S, there is exactly one vertex *v* such that $l(v) = a$.

The binary tree T in Fig. 3.38. is a binary search tree since every vertex in T exceeds every number in its left subtree and is less than every number in its right subtree.

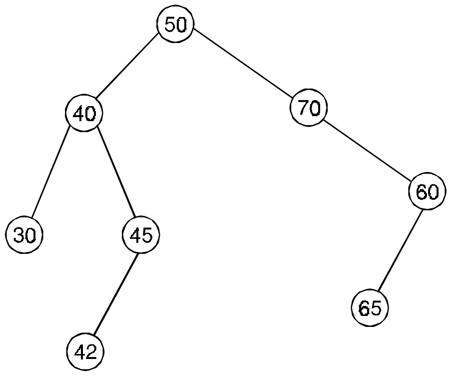


Fig. 3.38. A binary search tree.

3.9.1. Creating a binary search tree

The following recursive procedure is used to form the binary search tree for a list of items. To start, we create a vertex and place the first item in the list in this vertex and assign this as the key of the root. To add a new item, first compare it with the keys of vertices already in the tree. Starting at the root and moving to the left if the item is less than the key of the respective vertex if this has a left child, or moving to the right if the item is greater than the key of the respective vertex if this vertex has a right child when the item is less than the respective vertex and this vertex has no left child, then a new vertex with this item as its key is inserted as a new left child.

Similarly, when the item is greater than the respective vertex and this vertex has no right child, then a new vertex with this item as its key is inserted as a new right child. In this way, we store all the items in the list in the tree and thus create a binary search tree.

3.10. STORAGE REPRESENTATION OF BINARY TREE

In this section, we discuss two ways of representing a binary tree in computer memory. The first way uses a single array, called the sequential representation of binary tree. The second is called the link representation.

3.10.1. Sequential Representation

We can represent the vertices of a binary tree as array elements and access the vertices using array notations. The advantage is that we need not use a chain of pointers connecting the widely separated vertices.

Consider the almost complete binary tree shown in Fig. 3.39.

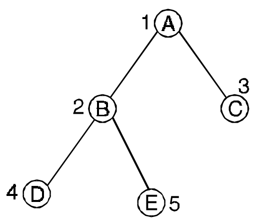


Fig. 3.39.

Note that we assigned numbers for all the nodes. We can assign numbers in such a way that the root is assigned the number 1, a left child is assigned twice the number assigned to its father, a right child is assigned one more than twice the number assigned to its father. We can keep the vertices of an almost complete binary tree in an array Fig. (3.40) shows vertices kept in an array.

Arrray positions vertices

1	2	3	4	5
A	B	C	D	E

Fig. 3.40.

By this convention, we can map vertex i to ith index in the array, and the parent of vertex i will get mapped at an index $i/2$ where as left child of vertex i gets mapped at an index $2i$ and right child gets mapped at an index $2i + 1$. The sequential representation can be extended to general binary trees. We do this by identifying an almost complete binary tree that contains the binary tree being represented. An almost complete binary tree containing the binary tree in Fig. (3.41) is shown in Fig. (3.42).

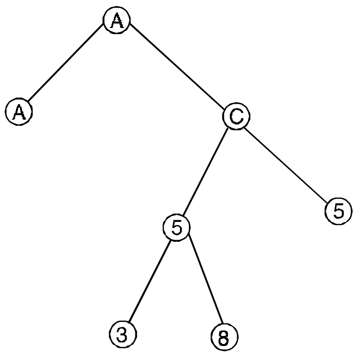


Fig. 3.41. A binary tree.

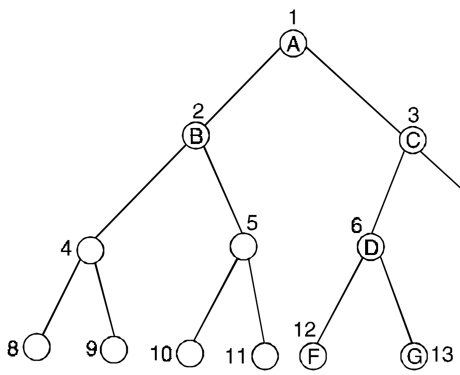


Fig. 3.42. An almost complete binary tree containing the binary tree in Fig. 3.41.

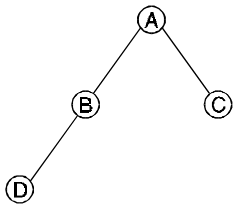
3.10.2. Linked representation

An array representation of a binary tree is not suitable for frequent insertions and deletions, and therefore we find that even though no storage is wasted if the binary tree is a complete one when array representation is used, it makes insertion and deletion is a tree is costly.

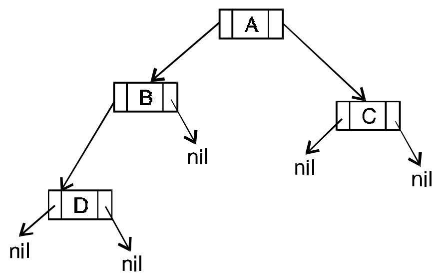
Computer representation of trees based on linked allocation seems to more popular because of the ease with which nodes can be inserted in and deleted from a tree, and because tree structure can grow to an arbitary size. Therefore instead of using an array representation, one can use a linked representation, in which every node is represented as a structure having 3 fields, one for holding data, one for linking it with left sub-tree and the one for linking it with right sub-tree as shown below. A general tree can easily be converted into an equivalent binary tree by using the natural correspondence algorithm.

Where LLINK or RLINK contain a pointer to the left sub-tree respectively of the node in question. DATA contains the information which is to be associated with this particular node. Each pointer can have a value of NULL.

An example of binary tree as a graph and its corresponding linked representation in memory are given in Fig. (3.43).



(*a*) : Binary tree



(*b*) : Linked representation of binary tree.

Fig. 3.43.

Problem 3.41. *Form a binary search tree*

(*i*) *for the data 16, 24, 7, 5, 8, 20, 40, 3 in the given order*

(*ii*) *for the words if, then, end, begin, else (used as keywords in ALGOL) in lexicographic order.*

Solution. (*i*) We begin by selecting the number 16 to be the root. Since the next number 24 is greater than 16, and a right child of the root and level it with 24.

We choose next element in the list 7 and again start at the root and compare it with 16. Since 7 is less than 16. Add a left child of the root and level it with 7.

We compare 5 to 7, since 7 is greater than 5, then we move further down to the right child of 7 and level the vertex to 5.

Similar procedure is followed for left out numbers in the list. The Fig. (3.44) shows the steps used to construct the binary search tree.

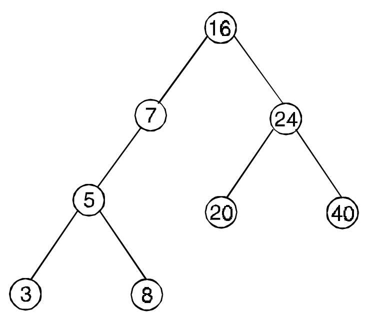


Fig. 3.44.

(*ii*) We start the word if as the key of the root. Since then comes after if (in alphabetical order), add a right child of the root with key then. Since the next word end comes before if, add a left child of the root with key end.

The next word begin is compared with if. Since begin is before if we move down to the left child of if, which is the vertex labelled end. We compare end with begin. Since begin comes before end, then we move further down to the left child of end and level with key begin.

Similarly else comes after begin, we move further down to the right child of begin and level with key else. The Fig. (3.45) shows the steps used to construct the binary search tree.

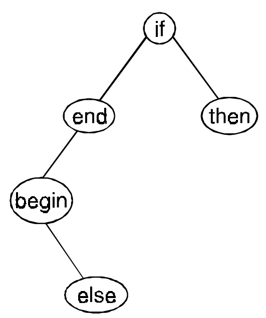


Fig. 3.45.

Problem 3.42. *Given the preorder and inorder traversal of a binary tree, draw the unique tree*

Preorder : g b q a c p d e r

Inorder : q b c a g p e d r

Solution. Here g is the first vertex in preorder traversal, thus g is the root of the tree.

Using inorder traversal, left subtree of g consists of the vertices q, b, c and a.

Then the left child g is b since b is the first vertex in the preorder traversal in the left subtree. Similarly, right subtree of g consists of the vertices P, e, d and r, then the right child of g is P since P is the vertex in the preorder traversal in the right subtree.

Repeating the above process with each node, we finally obtain the required tree as shown in Fig. (3.46).

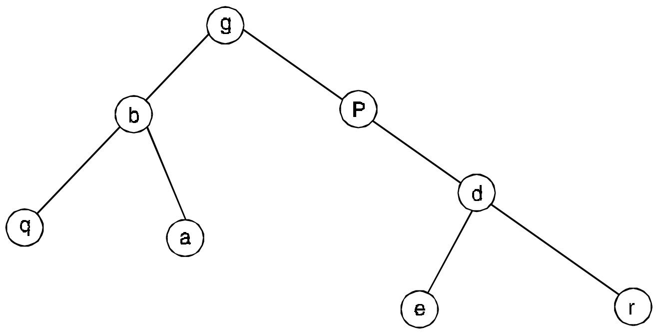


Fig. 3.46.

Problem 3.43. *Given the postorder and inorder traversal of a binary tree, draw the unique binary tree*

Post order : *d e c f b h i g a*

In order : *d c e b f a h g i*

Solution. Here *a* is the last vertex in postorder traversal, thus *a* is the root of the tree.

Using inorder traversal, right subtree of root vertex a consists of the vertices *h*, *g* and *i*.

The right child of *a* is *g* since *g* is the last vertex in the post order traversal in the right subtree.

Similarly, left subtree of *a* consists of the vertices *d*, *c*, *e*, *b* and *f* then the left child of *a* is *b* since *b* is the last vertex in the postorder traversal in the left subtree.

Repeating the above process with each vertex, we finally obtain the required tree as shown in Fig. (3.47).

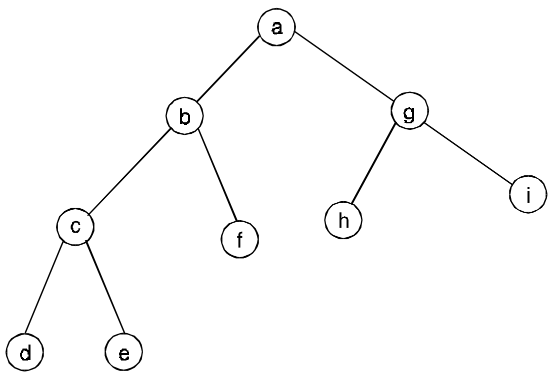


Fig. 3.47.

Problem 3.44. *Determine the value of the expression represented in a binary tree shown in Fig. (3.48).*

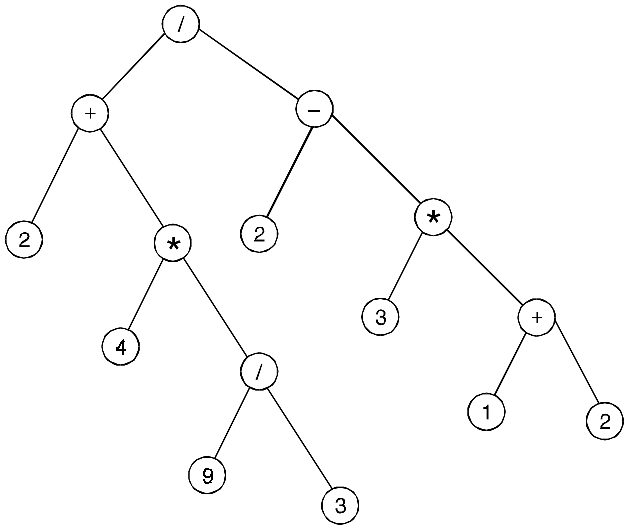


Fig. 3.48.

Solution. The expression represented by the binary tree is (9/3 × 4 + 2) / (((1 + 2) * 3) – 2) and the value is (3 * 4 + 2) / ((3 * 3) * 2) = (12 + 2) / (9 – 2) = 14/7 = 2.

Problem 3.45. *Use a binary tree to represent the expression*

(*i*) *a* * *b* (*ii*) (*a* + *b*)/*c* (*iii*) (*a* + *b*) * (*c*/*d*) (*iv*) ((*a* + *b*) * *c*) + (*d*/*e*)

Solution. (*i*)

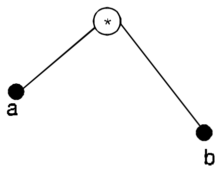


Fig. 3.49.

(*ii*)

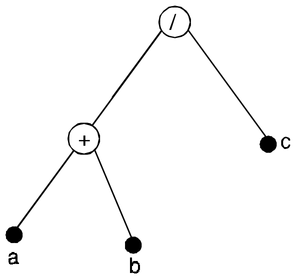


Fig. 3.50.

(*iii*)

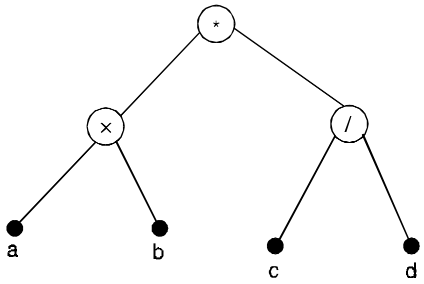


Fig. 3.51.

(*iv*)

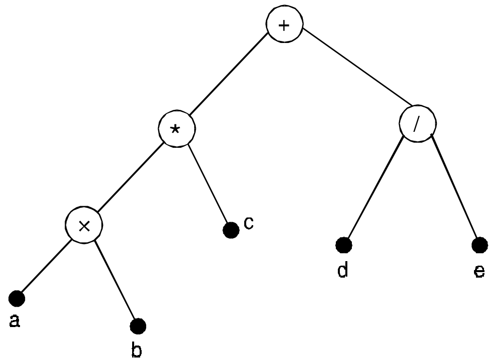


Fig. 3.52.

Theorem 3.25. *Let T = (V, E) be a complete m-ary tree of height h with l leaves. Then* $l \le m^h$ *and* $h \ge [\log_m l]$.

Proof. The proof that $l \le m^h$ will be established by induction on h.

When $h = 1$, T is a tree with a root and m children.

In this case $l = m = m^h$, and the result is true. Assume the result true for all trees of height $< h$, and consider a tree T with height h and l leaves. (the level numbers that are possible for these leaves are 1, 2, ..., h, with at least m of the leaves at level h).

The l leaves of T are also the l leaves (total) for the m subtree T_i, $1 \le i \le m$, of T rooted at each of the children of the root.

For $1 \le i \le m$, let l_i be the number of leaves in subtree T_i.

(In the case where leaf and root coincide, $l_i = 1$. But since $m \ge 1$ and $h - 1 \ge 0$, we have $m^{h-1} \ge 1 = l_i$)

By the induction hypothesis, $l_i \le m^{h(T_i)} \le m^{h-1}$, where $h(T_i)$ denotes the height of the subtree T_i, and so $l = l_1 + l_2 + \ldots\ldots + l_m \le m(m^{h-1}) = m^h$

With $l \le m^h$, we find that $\log_m l \le \log_m (m^h) = h$, and since $h \in Z^+$, it follows that $h \ge [\log_m l]$.

Corollary :

Let T be a balanced complete m-ary tree with l leaves. Then the height of T is $[\log_m l]$.

Problem 3.46. *Answer the following questions for the tree shown in Fig. (3.53).*

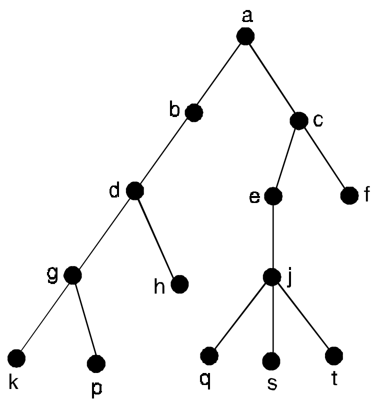


Fig. 3.53.

(*a*) *Which vertices are the leaves ?* (*b*) *Which vertex is the root ?*

(*c*) *Which vertex is the parent of g ?* (*d*) *Which vertices are the descendants ?*

(*e*) *Which vertices are the siblings of s ?* (*f*) *What is the level number of vertex f ?*

(*g*) *Which vertices have level number 4 ?*

Solution. (*a*) *f, h, k, p, q, s, t* (*b*) *a*

(*c*) *d* (*d*) *e, f, j, q, s, t*

(*e*) *q, t* (*f*) 2

(*g*) *k, p, q, s, t.*

Problem 3.47. *For the tree shown in Fig. (3.54), list the vertices according to a preorder traversal, an in order traversal, and a post order traversal.*

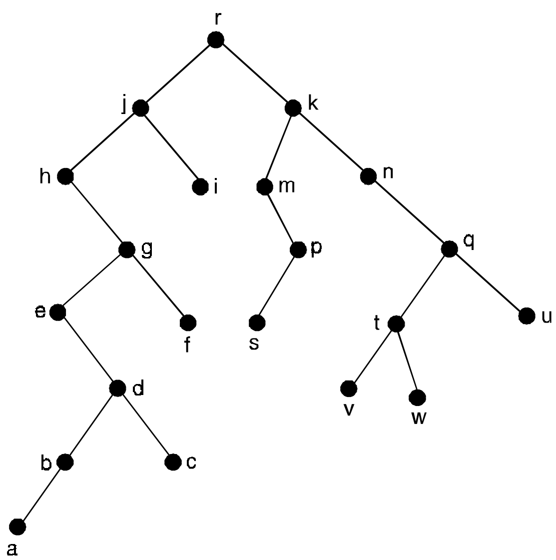


Fig. 3.54.

Solution. Preorder : *r, j, h, g, e, d, b, a, c, f, i, k, m, p, s, n, q, t, v, w, u*

Inorder : *h, e, a, b, d, c, g, f, j, i, r, m, s, p, k, n, v, t, w, q, u*

Postorder : *a, b, c, d, e, f, g, h, i, j, s, p, m, v, w, t, u, q, n, k, r*.

3.11. 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).

3.11.1. 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.

3.11.2. 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).*

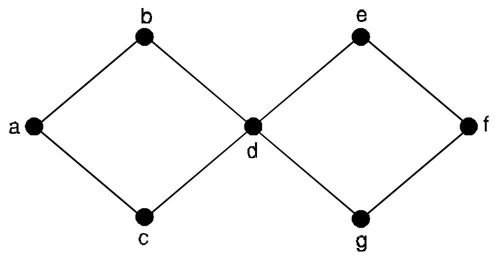


Fig. 3.55.

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.

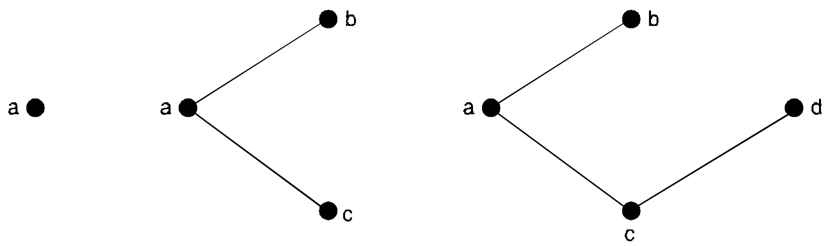
(*iii*) Add edges from these vertices at level 1 to adjacent vertices not already in the tree.

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).



(*a*) (*b*) (*c*)

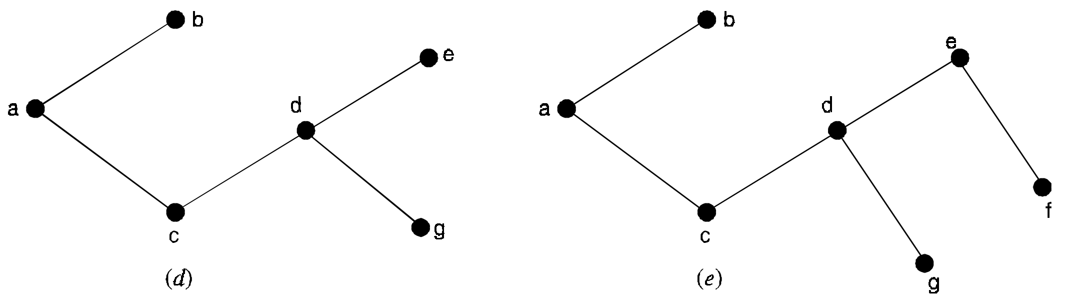


Fig. 3.56.

Hence, Fig. (3.56) (*e*) is the required spanning tree.

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

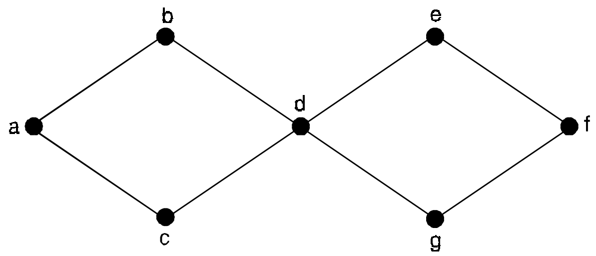


Fig. 3.57.

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).

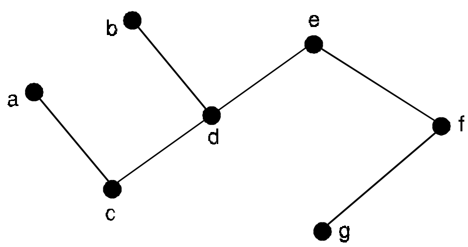


Fig. 3.58.

Problem 3.50. *Give all the spanning trees of K_4.*

Solution. Here $n = 4$, so there will be $4^{4-2} = 16$ different spanning trees.

All the spanning trees of K_4 are shown in Fig. (3.59).

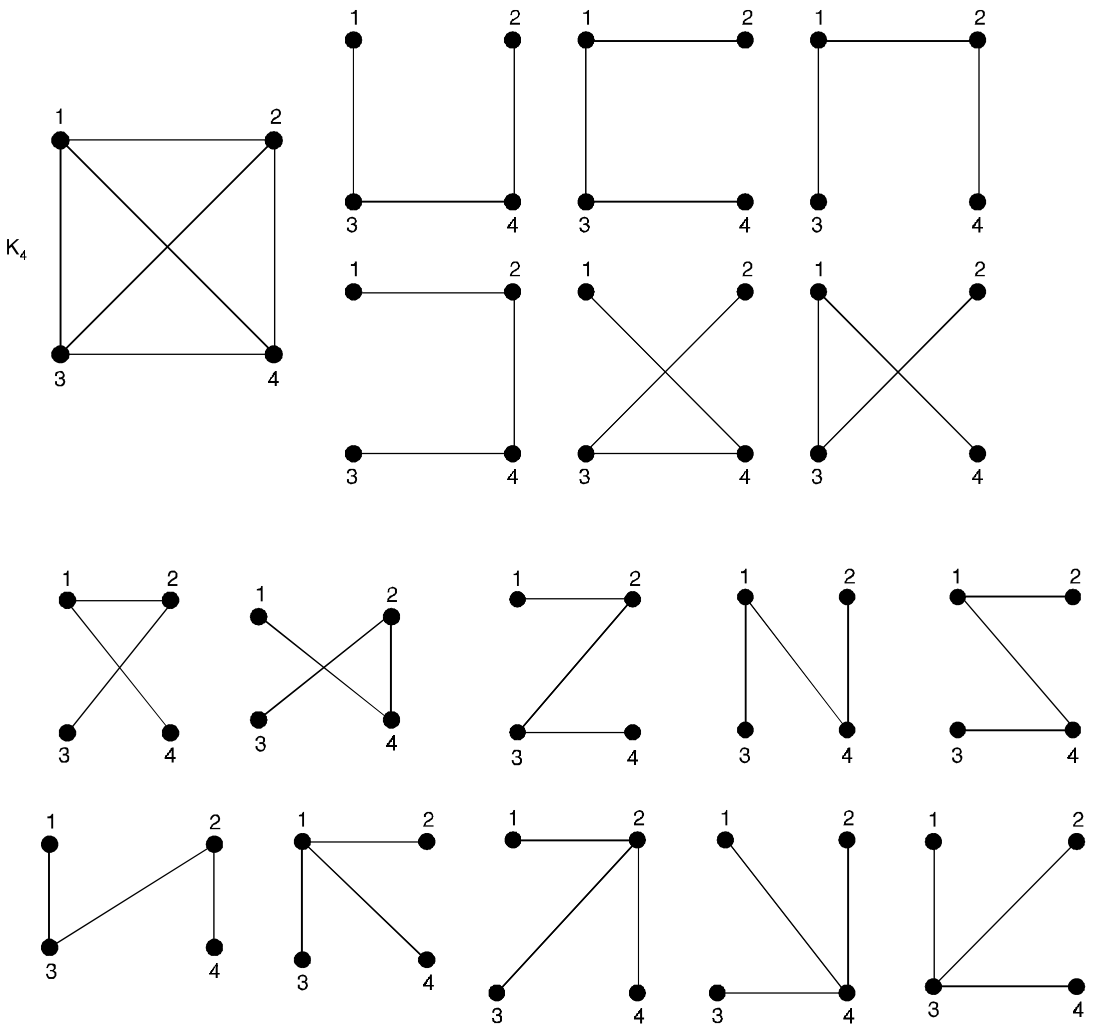


Fig. 3.59. K_4 and its 16 different spanning trees.

Theorem 3.26. *If G = (V, E) is an undirected graph, then G is connected if and only if G has a spanning tree.*

Proof. If G has a spanning tree T, then for every pair *a*, *b* of distinct vertices in V a subset of the edges in T provides a (unique) path between *a* and *b*, and so G is connected.

Conversely, if G is connected and G is not a tree, remove all loops from G.

If the resulting subgraph G_1 is not a tree, then G_1 must contain a cycle C_1.

Remove an edge e_1 from C_1 and let $G_2 = G_1 - e_1$.

If G_2 contains no cycles, then G_2 is a spanning tree for G because G_2 contains all the vertices in G, is loop-free, and is connected.

If G_2 does not contain a cycle, say, C_2, then remove and edge e_2 from C_2 and consider the subgraph $G_3 = G_2 - e_2 = G_1 - \{e_1, e_2\}$.

Once again, if G_3 contains no cycles, then we have a spanning tree for G.

Otherwise we continue this procedure a finite number of additional times until we arrive at a spanning subgraph of G that is loop-free and connected and contains no cycles. (and, consequently, is a spanning tree for G).

Theorem 3.27. *If a, b are distinct vertices in a tree T = (V, E), then there is a unique path that connects these vertices.*

Proof. Since T is connected, there is at least one path in T that connects *a* and *b*.

If there were more, then from two such paths some of the edges would form a cycle.

But T has no cycles.

Theorem 3.28. *For every tree T = (V, E), if | V | ≥ 2, then T has at least two pendant vertices.*

Proof. Let $| V | = n \geq 2$.

We know that $| E | = n - 1$, so, if G = (V, E) is an undirected graph or multigraph then $\sum_{v \in V} \deg(v)$ $= 2 | E |$ it follows that $2(n-1) = 2 | E | = \sum_{v \in V} \deg(v)$.

Since T is connected, we have $\deg(v) \geq 1$ for all $v \in V$.

If T has fewer than two pendant vertices, then either $\deg(v) \geq 2$ for all $v \in V$ or $\deg(v^*) = 1$ for only one vertex v^* in V.

In the first case we arrive at the contradiction

$$2(n-1) = \sum_{v \in V} \deg(v) \geq 2 | V | = 2n.$$

For the second case we find that

$$2(n-1) = \sum_{v \in V} \deg(v) \geq 1 + 2(n-1)$$

another contradiction.

Theorem 3.29. *If every tree T = (V, E), | V | = | E | + 1.*

Proof. The proof is obtained by applying the alternative form of mathematical induction to | E |.

If | E | = 0, then the tree consists of a single isolated vertex as in Fig. (3.60) (*a*).

Here | V | = 1 = | E | + 1, parts (*b*) and (*c*) of the figure verify the result for the cases, where | E | = 1 or 2.

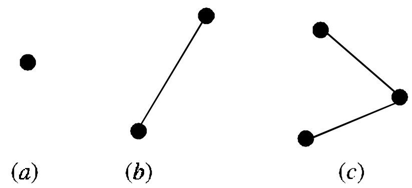


Fig. 3.60.

Assume the theorem is true for every tree that contains atmost k edges, where $k \geq 0$.

Now consider a tree T = (V, E), as in Fig. (3.61) where | E | = k + 1. (the dotted edges indicates that some of the tree does not appear in the figure).

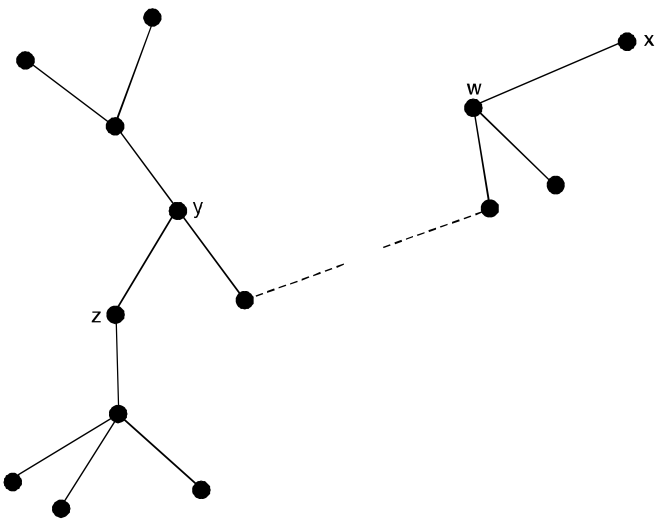


Fig. 3.61.

If, for instance, the edge with end points y, z is removed from T, we obtain two subtrees, $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$, where $| V | = | V_1 | + | V_2 |$ and $| E_1 | + | E_2 | + 1 = | E |$.

(One of the these subtrees could consists of just a single vertex if, for example, the edge with end points w, x were removed).

Since $0 \leq | E_1 | \leq K$ and $0 \leq | E_2 | \leq K$, it follows, by the induction hypothesis, that $| E_i | + 1 = | V_i |$, for $i = 1, 2$.

Consequently, $| V | = | V_1 | + | V_2 | = (| E_1 | + 1) + (| E_2 | + 1)$

$$= (| E_1 | + | E_2 | + 1) + 1$$

$$= | E | + 1,$$

and the theorem follows by the alternative form of mathematical induction.

Theorem 3.30. *The following statements are equivalent for a loop-free undirected graph G = (V, E).*

(*i*) *G is a tree*

(*ii*) *G is connected, but the removal of any edge from G disconnects G into two subgraphs that are trees.*

(*iii*) *G contains no cycles, and* $| V | = | E | + 1$

(*iv*) *G is connected, and* $| V | = | E | + 1$

(*v*) *G contains no cycles, and if* $a, b \in V$ *with* $\{a, b\} \notin E$ *then the graph obtained by adding edge* $\{a, b\}$ *to G has precisely one cycle.*

Proof. (*i*) $\Rightarrow$ (*ii*)

If G is a tree, then G is connected. So let $e = \{a, b\}$ be any edge of G.

Then if G – *e* is connected, there are at least two paths in G from *a* and *b*.

Hence G – *e* is disconnected and so the vertices in G – *e* may be partitioned into two subsets.

(1) Vertex *a* and those vertices that can be reached from *a* by a path in G – *e*, and

(2) Vertex *b* and those vertices that can be reached from *b* by a path in G – *e*.

There two connceted components are trees because a loop or cycle in either component would also be in G.

(*ii*) $\Rightarrow$ (*iii*)

If G contains a cycle, then $e = \{a, b\}$ be an edge of the cycle. But then G – *e* is connected, contradicting the hypothesis in part (*ii*).

So G contains no cycles, and since G is a loop-free connected undirected graph, we know that G is a tree.

Consequently, it follows from | V | = | E | + 1.

(*iii*) $\Rightarrow$ (*iv*)

Let K(G) = *r* and let $G_1, G_2, \ldots, G_r$ be the components of G.

For $1 \le i \le r$, select a vertex $v_i \in G_i$ and add the $r - 1$ edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{r-1}, v_r\}$ to G to form the graph G′ = (V, E′), which is a tree.

Since G′ is a tree, we know that | V | = | E′ | + 1 because of, in every tree T = (V, E), | V | = | E | + 1.

But from part (*iii*), | V | = | E | + 1, so | E | = | E′ | and $r - 1 = 0$, with $r = 1$, it follows that G is connected.

Problem 3.51. *If G = (V, E) is a loop-free undirected graph, prove that G is a tree if there is a unique path between any two vertices of G.*

Solution. If there is a unique path between each pair of vertices in G, then G is connected.

If G contains a cycle, then there is a pair of vertices *x, y* with two distinct paths connecting *x* and *y*.

Hence, G is a loop-free connected undirected graph with no cycles, so G is a tree.

Problem 3.52. *Let G = (V, E) be a loop-free connected undirected graph. Let H be a subgraph of G. The complement of H in a G is the subgraph of G made up of those edges in G that are not in H (along with the vertices incident to these edges).*

(*i*) *If T is a spanning tree of G, prove that the complement of T in G does not contain a cut-set of G.*

(*ii*) *If C is a cut-set of G, prove that the complement of C in G does not contain a spanning tree of G.*

Solution. (*i*) If the complement of T contains a cut set, then the removal of these edges disconnects G, and these are vertices *x, y* with no path connecting them. Hence T is not a spanning tree for G.

(*ii*) If the complement of C contains a spanning tree, then every pair of vertices in G has a path connecting them, and this path includes no edges of C.

Hence the removal of the edges in C from G does not disconnect G, so C is not a cut set for G.

Problem 3.53. *A labeled tree is one where in the vertices are labeled. If the tree has n vertices, then {1, 2, 3, ..., n} is used as the set of labels. We find that two trees that are isomorphic without labels may become non isomorphic when labeled. In Fig. (3.62) the first two trees are isomorphic as labelled trees. The third tree is isomorphic to the other two if we ignore the labels, as a labeled tree, however, it is not isomorphic to either of the other two.*

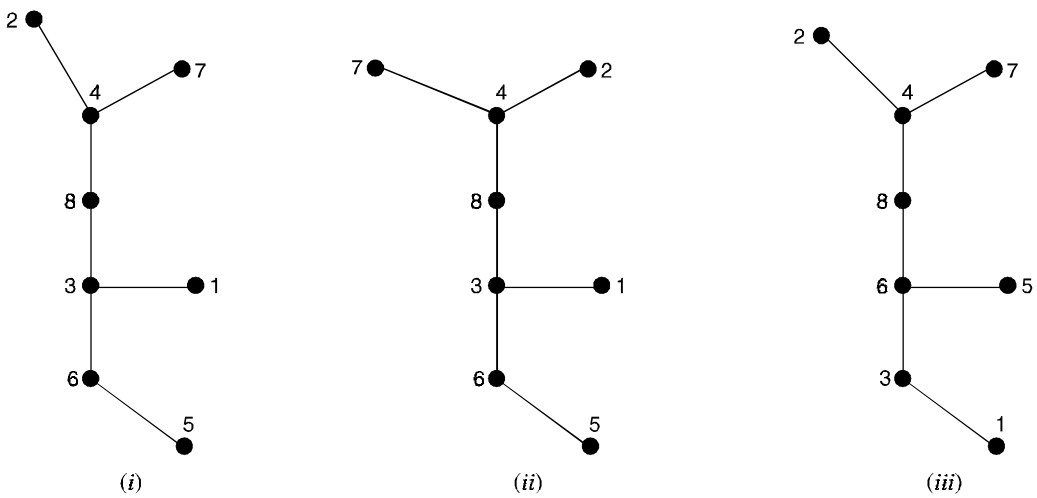


Fig. 3.62.

The number of non isomorphic trees with n labeled vertices can be counted by setting up a one-to-one correspondence between these trees and the n^{n-2} *sequence (with repetitions allowed)* $x_1, x_2, ...$ x_{n-2} *whose entries are taken from {1, 2, 3, ..., n}. If T is one such labeled tree, we use the following algorithm to establish the one-to-one correspondence. (Here T has at least one edge)*

Step 1 : set the counter i to 1

Step 2 : set T(i) = T

Step 3 : Since a tree has at least two pendant vertices, select the pendant vertex in T(i) with the smallest label y_i*. Now remove the edge* $\{x_i, y_i\}$ *from T(i) and use* x_i *for the* i^{th} *component of the sequence.*

Step 4 : If i = n – 2, we have the sequence corresponding to the given labeled tree T(1). If i ≠ n – 2, increase i by 1, set T(i) equal to the resulting subtree obtained in step (3), and return to step (3).

(*a*) *Find the six-digit sequence for trees (i) and (iii) in Fig. (3.62).*

(*b*) *If v is a vertex in T, show that the number of times the label on v appears in the sequence* $x_1, x_2, \ldots\ldots x_{n-2}$ *is deg (v) – 1.*

(*c*) *Reconstruct the labeled tree on eight vertices that is associated with the sequence 2, 6, 5, 5, 5, 5.*

(*d*) *Develop an algorithm for reconstructing a tree from a given sequence* $x_1, x_2, \ldots\ldots x_{n-2}$.

Solution. (*a*) (*i*) 3, 4, 6, 3, 84 (*ii*) 3, 4, 6, 6, 8, 4

(*b*) No pendant vertex of the given tree appears in the sequence, so the result is true for these vertices. When an edge {*x*, *y*} is removed and *y* is a pendant vertex (of the tree or one of the resulting subtrees), the deg (*x*) is decreased by 1 and *x* is placed in the sequence.

As the process continues, either (*i*) this vertex *x* becomes a pendant vertex in a subtree and is removed but not recorded again in the sequence, or (*ii*) the vertex *x* is left as one of the last two vertices of an edge.

In either case *x* has been listed in the sequence [deg (*x*) – 1] times.

(*c*)

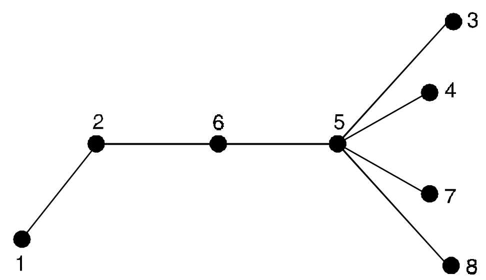


Fig. 3.63.

(*d*) From the given sequence the degree of each vertex in the tree is known.

Step 1 : Set the counter *i* to 1.

Step 2 : From among the vertices of degree 1, selected the vertex *v* with the smallest label.

This determines the edge $\{v, x_i\}$. Remove *v* from the set of labels and reduce the degree of x_i by 1.

Step 3 : If $i < n - 2$, increase *i* by 1 and return to step (2).

Step 4 : If $i = n - 2$, the vertices (labels) x_{n-3}, x_{n-2} are connected by an edge if $x_{n-3} \neq x_{n-2}$. (The tree is then complete).

Problem 3.54. *Let G = (V, E) be a loop-free undirected graph with | V | = n. Prove that G is a tree if and only if,* $P(G, \lambda) = \lambda(\lambda - 1)^{n-1}$.

Solution. If G is a tree, consider G a rooted tree. Then these are λ choices for coloring the root of G and (λ – 1) choices for coloring each of its descendants. The result then follows by rule of product.

Conversely, if $P(G, \lambda) = \lambda(\lambda - 1)^{n-1}$, then since the factor λ occurs only once, the graph G is connected.

$$P(G, \lambda) = \lambda(\lambda - 1)^{n-1} = \lambda^n - (n-1)\lambda^{n-1} + ... + (-1)^{n-1}\lambda$$

$\Rightarrow$ G has n vertices and $(n-1)$ edges.

Hence G is a tree.

Problem 3.55. *Let G = (V, E) be a loop-free undirected graph. If deg (v) ≥ 2 for all v ∈ V, prove that G contains a cycle.*

Solution. We assume that G = (V, E) is connected, otherwise we work with a component of G.

Since G is connected, and deg (v) ≥ 2 for all $v \in$ V, it follows from, for every tree T = (V, E), if | V | ≥ 2, then T has at least two pendant vertices, that G is not a tree. But every loop-free connected undirected graph that is not a tree must contain a cycle.

Problem 3.56. *Let T = (V, E) be a tree with V = {v_1, v_2, ... v_n}, for n ≥ 2. Prove that the number of pendant vertices in T is equal to* $2 + \sum_{deg(v_i) \geq 3} (deg(v_i) - 2)$.

Solution. For $1 \leq i$ $(< n)$, let x_i = the number of vertices v, where deg (v) = i.

Then $x_1 + x_2 + ... + x_{n-1} = |V| = |E| + 1$, so $2|E| = 2(-1 + x_1 + x_2 + ... + x_{n-1})$

But $2|E| = \sum_{v \in V} \deg(v) = (x_1 + 2x_2 + 3x_3 + ... + (n-1)x_{n-1})$

Solving $2(-1 + x_1 + x_2 + ... + x_{n-1}) = x_1 + 2x_2 + ... + (n-1)x_{n-1}$ for x_1, we find that

$$x_1 = 2 + x_3 + 2x_4 + 3x_5 + ... + (n-3)x_{n-1}$$

$$= 2 + \sum_{\deg(v_i) \geq 3} [\deg(v_i) - 2].$$

Problem 3.57. *Suppose that some one starts a chain letter. Each person who receives the letter is asked to send it on to four other people. Some people do this, but others do not send any letters. How many people have seen the letter, including the first person, if no one receives more than one letter and if the chain letter ends after there have been 100 people who read it but did not send it out ? How many people sent out the letter ?*

Solution. The chain letter can be represented using a 4-ary tree.

The internal vertices correspond to people who sent out the letter, and the leaves correspond to people who did not send it out.

Since 100 people did not send out the letter, the number of leaves in this rooted tree is $l = 100$.

Hence, the number of people who have seen the letter is $n = (4 \,.\, 100 - 1)/(4 - 1) = 133$.

Also, the number of internal vertices is 133 – 100 = 33.

So, that 33 people sent out the letter.

Problem 3.58. *Which of the rooted trees shown in Fig. (3.64) are balanced ?*

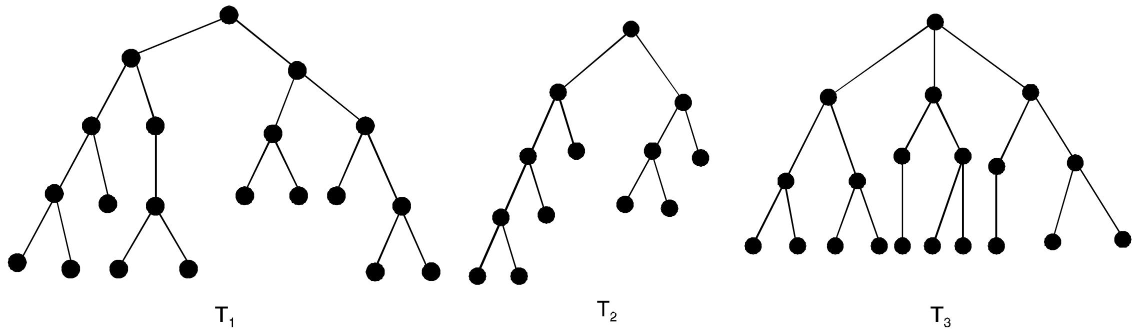


Fig. 3.64.

Solution. T_1 is balanced, since all its leaves are at levels 3 and 4.

However, T_2 is not balanced, since it has leaves at levels 2, 3 and 4.

Finally, T_3 is balanced, since all its leaves are at level 3.

Problem 3.59. *Find the level of each vertex in the rooted tree shown in Fig. (3.65). What is the height of this tree ?*

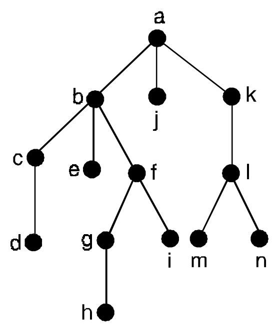


Fig. 3.65.

Solution. The root *a* is at level 0. Vertices *b*, *j* and *k* are at level 1. Vertices *c*, *e*, *f* and *l* are at level 2. Vertices *d*, *g*, *i*, *m* and *n* are at level 3.

Finally, vertex *h* is at level 4. Since the largest level of any vertex is 4, this tree has height 4.

Problem 3.60. *Which of the graphs shown in Fig. (3.66) are trees ?*

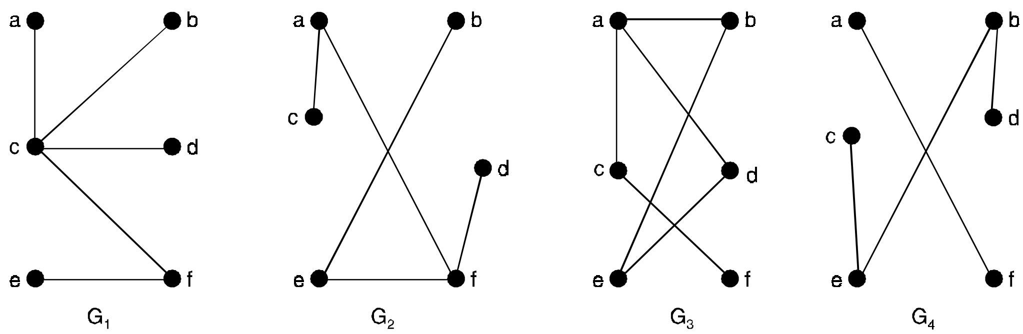


Fig. 3.66.

Solution. G_1 and G_2 are trees, since both are connected graphs with no simple circuits. G_3 is not a tree because *e*, *b*, *a*, *d*, *e* is a simple circuit in this graph.

Finally, G_4 is not a tree since it is not connected.

Theorem 3.31. *An undirected graph is a tree if and only if there is a unique simple path between any two of its vertices.*

Proof. First assume that T is a tree. Then T is a connected graph with no simple circuits.

Let *x* and *y* be two vertices of T. Since T is connected there is a simple path between *x* and *y*.

Moreover, this path must be unique, for if there were a second such path, the path formed by combining the first path from *x* to *y* followed by the path from *y* to *x* obtained by reversing the order of the second path from *x* to *y* would form a circuit. This implies that there is a simple circuit in T.

Hence, there is a unique simple path between any two vertices of a tree.

Now assume that there is a unique simple path between any two vertices of a graph T.

Then T is connected, since there is a path between any two of its vertices.

Furthermore, T can have no simple circuits. To see that this is true, suppose T has a simple circuit that contained the vertices *x* and *y*.

Then there would be two simple paths between *x* and *y*, since the simple circuit is made up of a simple path from *x* to *y* and a second simple path from *y* to *x*.

Hence, a graph with a unique simple path between any two vertices is a tree.

Problem 3.61. *In the rooted tree T (with root a) shown in Fig. (3.67), find the parent of C, the children of g, the siblings of h, all ancestors of e, all descendants of b, all internal vertices, and all leaves. What is the subtree rooted at g ?*

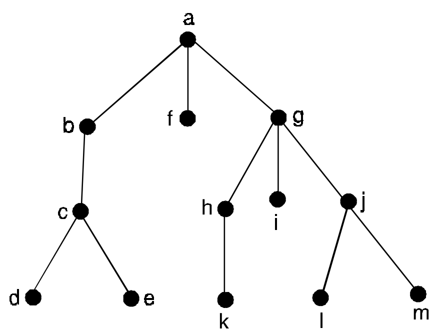


Fig. 3.67. A rooted tree T.

Solution. The parent of *c* is *b*. The children of *g* are *h*, *i* and *j*. The siblings of *h* are *i* and *j*.

The ancestors of *e* are *c*, *b* and *a*. The descendants of *b* are *c*, *d* and *e*.

The internal vertices are *a*, *b*, *c*, *g*, *h* and *j*.

The leaves are *d*, *e*, *f*, *i*, *k*, *l* and *m*.

The subtree rooted at *g* is shown in Fig. (3.68).

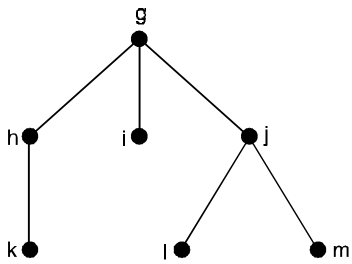


Fig. 3.68. The subtree rooted at *g*.

Problem 3.62. *Are the rooted trees in Fig. (3.69) full m-ary trees for some positive integer m ?*

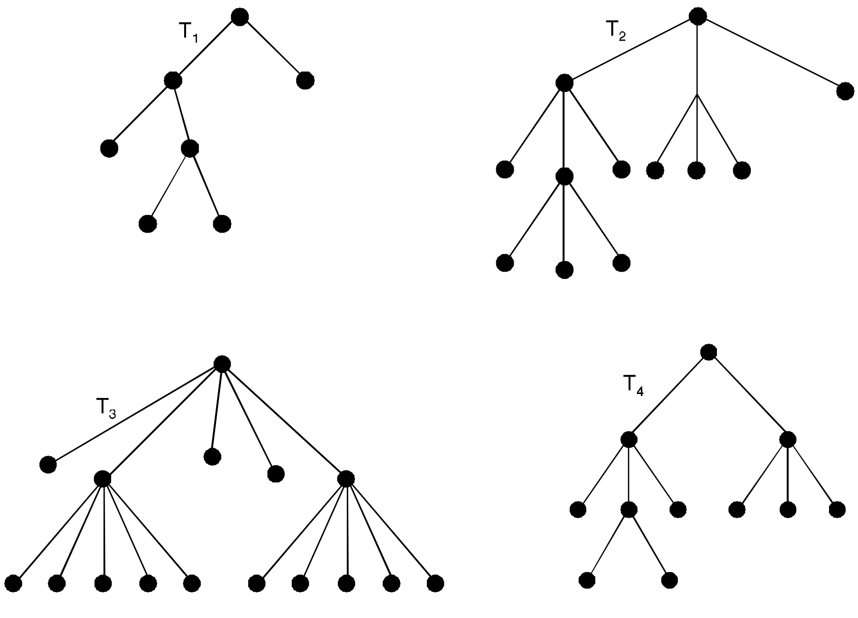


Fig. 3.69. Four rooted trees.

Solution. T_1 is a full binary tree since each of its internal vertices has two children.

T_2 is a full 3-ary tree since each of its internal vertices has three children.

In T_3 each internal vertex has five children, so T_3 is a full 5-ary tree.

T_4 is not a full *m*-ary tree for any *m* since some of its internal vertices have two children and others have three children.

Problem 3.63. *What are the left and right children of d in the binary tree T shown in Fig. (3.70). (Where the order is that implied by the drawing). What are the left and right subtrees of C ?*

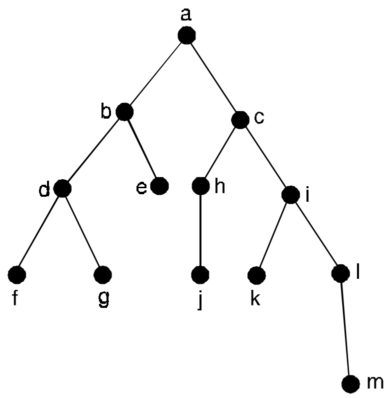
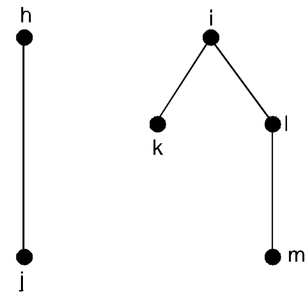


Fig. 3.70.

Solution. The left child of d is f and the right child is g. We show the left and right subtrees of c in Fig. (3.71) (a) and (b), respectively.



(a) (b)

Fig. 3.71.

3.12. TREES AND SORTING

3.12.1. Decision trees

Rooted trees can be used to model problems in which a series of decisions leads to a solution. For instance, a binary search tree can be used to locate items based on a series of comparisons, where each comparison tells us whether we have located the item, or whether we should go right or left in a subtree.

A rooted tree in which each internal vertex corresponds to a decision, with a subtree at these vertices for each possible outcome of the decision, is called a **decision tree.** The possible solutions of the problem correspond to the paths to the leaves of this rooted tree.

3.12.2. The complexity of sorting algorithms

Many different sorting algorithms have been developed. To decide whether a particular sorting algorithm is efficient, its complexity is determined. Using decision trees as models, a lower bound for the worst-case complexity of sorting algorithms can be found. We can use decision trees to model sorting algorithms and to determine an estimate for the worst-case complexity of these algorithms.

Note that given n elements, there are n ! possible orderings of these elements, since each of the n ! permutations of these elements can be the correct order. The sorting algorithms are based on binary comparisons, that is, the comparison of two elements at a time. The result of each such comparison narrows down the set of possible orderings.

Thus, a sorting algorithms based on binary comparisons can be represented by a binary decision tree in which each internal vertex represents a comparison of two elements. Each leaf represents one of the n ! permutations of n elements.

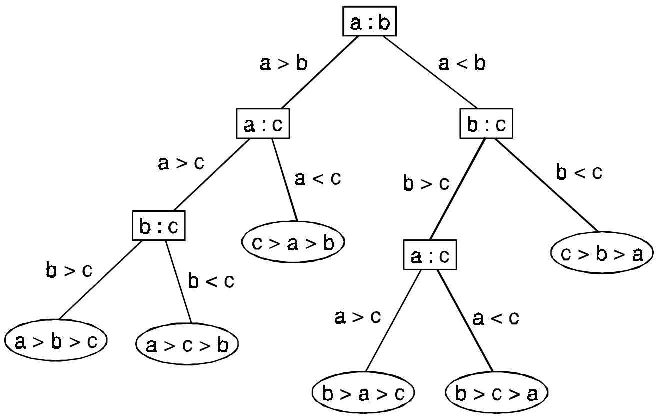


Fig. 3.72. A Decision tree for sorting three distinct elements.

3.12.3. The merge sort algorithms

For sorting a given list of n items into ascending order. The method is called the merge sort, and we find that the order of its worst-case time-complexity function is 0/(n $\log_2 n$). This will be accomplished in the following manner.

(*i*) First we measure the number of comparisons needed when n is a power of 2. Our method will apply a pair of balanced complete binary trees.

(*ii*) Then we cover the case for general n by using optional material on divide and conquer alorithms, where

(*a*) The time to solve the intial problem of size $n = 1$ is a constant $c \geq 0$ (solving the problem for a small value of 'n' directly).

(*b*) The time to break the given problem of size n into a smaller (similar) problems, together with the time to combine the solutions of these smaller problems to get a solution for the given problem, is $h(n)$, a function of n.

For the case where n is an arbitrary positive integer, we start by considering the following procedure.

Given a list of n items to sort into ascending order, the merge sort recursively splits the given list and all subsequent sublists in half until each sublist contains a single element. Then the procedure merges these sublists in ascending order until the original n items have been so sorted. The splitting and merging processes can best be described by a pair of balanced complete binary trees.

Merge sort algorithm

Procedure :

Step 1 : If $n = 1$, then list is already sorted and the process terminates. If $n > 1$, then go to step (2).

Step 2 : (Divide the array and sort the subarrays). Perform the following :

(*i*) Assign m the value $\lfloor n/2 \rfloor$

(*ii*) Assign to List 1 the subarray

List [1], List [2], ... List [m].

(*iii*) Assign to List 2 the subarray

List [m + 1], List [m + 2], ..., List [n].

(*iv*) Apply merge sort to List 1 (of size m) and to List 2 (of size $n - m$).

Step 3 : Merge (List 1, List 2).

Problem 3.64. *Determine the (worst-case) time-complexity function of the merge sort, consider a list of n elements, assuming that* $n = 2^h$.

Solution. In the splitting process, the list of 2^h elements is first split into two sublists of size 2^{h-1}. These are the level 1 vertices in the tree representing the splitting process. As the process continues, each successive list of size 2^{h-k}, $h > k$, is at level k and splits into two sublists of size (1/2) $(2^{h-k}) = 2^{h-k-1}$. At level h the sublists each contain $2^{h-k} = 1$ element.

Reversing the process, we first merge the $n = 2^h$ leaves into 2^{h-1} ordered sublists of size 2.

These sublists are at level $h - 1$ and require (1/2) $(2^h) = 2^{h-1}$ comparisons.

As this merging process continues, at each of the 2^k vertices at level k, $1 \le k < h$, there is a sublist of size 2^{h-k}, obtained from merging the two sublists of size 2^{h-k-1} at its children (on level $k + 1$). This merging requires at most $2^{h-k-1} + 2^{h-k-1} - 1$

$$= 2^{h-k} - 1 \text{ comparisons.}$$

When the children of the root are reached, there are two sublists of size 2^{h-1} (at level 1).

To merge these sublists into the final list requires at most $2^{h-1} + 2^{h-1} - 1 = 2^h - 1$ comparisons.

Consequently, for $1 \le k \le h$, at level k there are 2^{k-1} pairs of vertices.

At each of these vertices is a sublist of size 2^{h-k}, so it takes atmost $2^{h-k+1} - 1$ comparisons to merge each pair of sublists.

With 2^{k-1} pairs of vertices at level k, the total number of comparisons at level k is atmost $2^{k-1}(2^{h-k+1} - 1)$.

When we sum over all levels k, where $1 \le k \le$ h, we find that the total number of comparisons is

at most $\displaystyle\sum_{k=1}^{h} 2^{k-1}\ (2^{h-k+1} - 1) = \sum_{k=0}^{h-1} 2^k\ (2^{h-k} - 1)$

$$= \sum_{k=0}^{h-1} 2^h - \sum_{k=0}^{h-1} 2^k$$

$$= h \,.\, 2^h - (2^h - 1)$$

With $n = 2^h$, we have $h = \log_2 n$ and

$$h \cdot 2^h - (2^h - 1) = n \log_2 n - (n - 1)$$
$$= n \log_2 n - n + 1.$$

Where $n \log_2 n$ is the dominating term for large n.

Thus the (worst-case) time-complexity function for this sorting procedure is $g(n) = n \log_2 n - n + 1$ and $g \in 0\ (n \log_2 n)$, for $n = 2^h$, $h \in z^+$.

Hence the number of comparisons needed to merge sort a list of n items is bounded above by $dn \log_2 n$ for some constant d, and for all $n \geq n_0$, where n_0 is some particular (large) positive integer.

Problem 3.65. *Show that the order of the merge sort is 0 (n $\log_2$ n) for all n ∈ z^+.*

Solution. Let $a, b, c \in z^+$, with $b \geq 2$.

If $g : z^+ \longrightarrow R^+ \cup \{0\}$ is a monotone increasing function, where $g(1) \leq c$,

$g(n) \leq ag\ (n/b) + cn$, for $n = b^h, h \in Z^+$, then for the case where $a = b$, we have $g \in 0\ (n \log n)$ for all $n \in Z^+$.

The function $g : z^+ \longrightarrow R^+ \cup \{0\}$ will measure the (worst-case) time-complexity for this algorithm by counting the maximum number of comparisons needed to merge sort an array of n items.

For $n = 2^h$, $h \in z^+$, we have

$$g(n) = 2 \log (n/2) + [(n/2) + (n/2) - 1]$$

The term $2g\ (n/2)$ results from step (2) of the merge sort algorithm, and the summand $[(n/2) + (n/2) - 1]$ follows from step (3) of the algorithms.

With $g(1) = 0$, the preceding equation provides the in equalities $g(1) = 0 \leq 1$.

$$g(n) = 2g\ (n/2) + (n - 1) \leq 2g\ (n/2) + n$$

for $n = 2^h$, $h \in Z^+$.

We also observe that $g(1) = 0$, $g(2) = 1$, $g(3) = 3$ and $g(4) = 5$, so $g(1) \leq g(2) \leq g(3) \leq g(4)$.

Consequently, it appears that g may be a monotone increasing function.

Lemma :

Let L_1 and L_2 be two sorted lists of ascending numbers. Where L_i contains n_i elements, for $i = 1, 2$. Then L_1 and L_2 can be merged into one ascending list L using atmost $n_1 + n_2 - 1$ comparisons.

Proof. To merge L_1, L_2 into list L, we perform the following algorithm.

Step 1 : Set L equal to the empty list ϕ

Step 2 : Compare the first elements in L_1, L_2.

Remove the smaller of the two from the list it is in and place it at the end of L.

Step 3 : For the present lists L_1, L_2 (one change is made in one of these lists each time step (2) is executed), there are two considerations.

(*a*) If either of L_1, L_2 is empty, then the other list is concatenated to the end of L. This completes the merging process.

(*b*) If not, return to step (2).

Each comparison of a number from L_1 with one from L_2 results in the placement of an element at the end of list L, so there cannot be more than $n_1 + n_2$ comparisons.

When one of the lists L_1 or L_2 becomes empty no further comparisons are needed, so the maximum number of comparisons needed is $n_1 + n_2 - 1$.

Theorem 3.32. *A sorting algorithm based on binary comparisons requires at least [log n !] comparisons.*

Corollary. The number of comparisons used by a sorting algorithm to sort *n* elements based on binary comparisons in Ω (*n* log *n*).

Theorem 3.33. *The average number of comparisons used by a sorting algorithms to sort n elements based on binary comparisons is* Ω *(n log n).*

Problem 3.66. *Suppose there are seven coins, all with the same weight, and a counterfeit coin that weighs less than the others. How many weighings are necessary using a balance scale to determine which of the eight coins is the counterfeit one ? Give an algorithm for finding this counterfeit coin.*

Solution. There are three possibilities for each weighing on a balance scale.

The two pans can have equal weight, the first pan can be heavier, or the second pan can be heavier.

Consequently, the decision tree for the sequence of weighings is a 3-ary tree.

There are at least eight leaves in the decision tree since there are eight possible outcomes (since each of the eight coins can be the counterfeit lighter coin), and each possible outcome must be represented by at least one leaf.

The largest number of weighings needed to determine the counterfeit coin is the height of the decision tree. The height of the decision tree is at least [$\log_3 8$] = 2.

Hence, at least two weighings are needed.

It is possible to determine the counterfeit coin using two weighings.

Problem 3.67. *Using the Merge Sort, sorts the list 6, 2, 7, 3, 4, 9, 5, 1, 8.*

Solution. The tree at the top of the Figure (3.73) shows how the process first splits the given list into sublists of SIZE 1.

The merging process is than outlined by the tree at the bottom of the Figure 3.73.

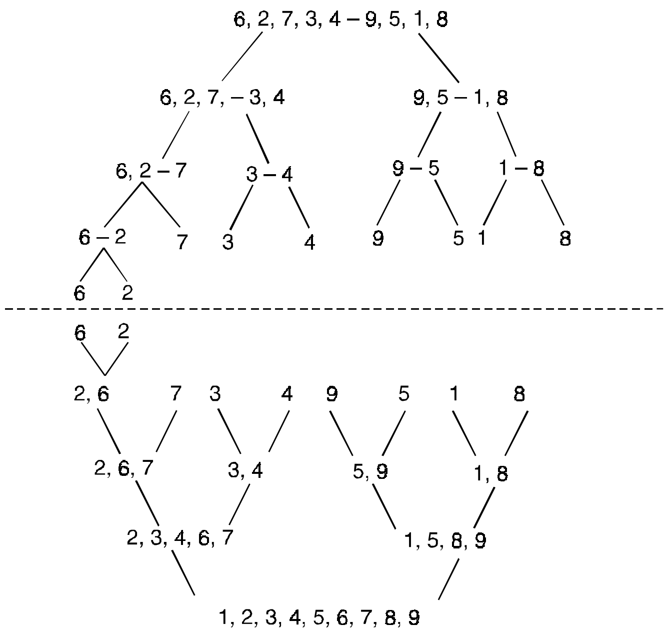


Fig. 3.73.

3.13. WEIGHTED TREES AND PREFIX CODES

Consider the problem of using bit strings to encode the letters of the English alphabet (where no distinction is made between lower case and upper case letters).

We can represent each letter with a bit string of length five, since there are only 26 letters and there are 32 bit strings of length five.

The total number of bits used to encode data is five times the number of characters in the text when each character is encoded with five bits. Is it possible to find a coding scheme of these letters so that, when data are coded, fewer bits are used ? We can save memory and reduce transmittal time if this can be done.

Consider using bit strings of different lengths to encode letters. Letters that occur more frequently should be encoded using short bit strings, and longer bit strings should be used to encode rarely occurring letters. When letters are encoded using varying numbers of bits, some method must be used to determine where the bits for each character start and end.

For instance, if *e* were encoded with 0, *a* with 1, and *t* with 01, then the bit string 0101 could correspond to *eat*, *tea*, *eaea*, or *tt*.

One way to ensure that no bit string corresponds to more than one sequence of letters is to encode letters so that the bit string for a letter never occurs as the first part of the bit string for another letter. Codes with this property are called **prefix codes.**

For instance, the encoding of *e* as 0, *a* as 10, and *t* as 11 is a prefix code.

A word can be recovered from the unique bit string that encodes its letters.

For example, the string 10110 is the encoding of *ate*.

To see this, note that the initial 1 does not represent a character, but 10 does represent a (and could not be the first part of the bit string of another letter).

Then, the next 1 does not represent a character but 11 does represent *t*. The final bit, 0, represents *e*.

A prefix code can be represented using a binary tree, where the characters are the labels of the leaves in the tree. The edges of the tree are labeled so that an edge leading to a left child is assigned a 0 and an edge leading to a right child is assigned a 1.

The bit string used to encode a character is the sequence of labels of the edges in the unique path from the root to the leaf that has this character as its label.

For instance, the tree in (3.74) represents the encoding of *e* by 0, *a* by 10, *t* by 110, *n* by 1110, and *s* by 1111.

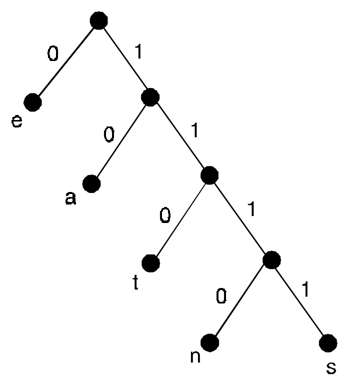


Fig. 3.74. The binary tree with a prefix code.

The tree representing a code can be used to decode a bit string. For instance, consider the word encoded by 11111011100 using the code in Fig. (3.74).

This bit string can be decoded by starting at the root, using the sequence of bits to form a path that stops when a leaf is reached. Each 0 bit takes the path down the edge leading to the left child of the last vertex in the path, and each 1 bit corresponds to the right child of this vertex.

Consequently, the initial 1111 corresponds to the path starting at the root, going right four times, leading to a leaf in the graph that has *s* as its label, since the string 1111 is the code for *s*.

Continuing with the fifth bit, we reach a leaf next after going right then left, when the vertex labeled with *a*, which is encoded by 10, is visited.

Starting with the seventh bit, we reach a leaf next after going right three times and then left, when the vertex labeled with *n*, which is encoded by 1110, is visited. Finally, the last bit, 0, leads to the leaf that is labeled with *e*. Therefore, the original word is same.

We can construct a prefix code from any binary tree where the left edge at each internal vertex is labeled by 0 and the right edge by a 1 and where the leaves are labeled by characters. Characters are encoded with the bit string constructed using the labels of the edges in the unique path from the root to the leaves.

3.13.1. Huffman coding

We now introduce an algorithm that takes as input the frequencies (which are the probabilities of occurrences) of symbols in a string and produces as output a prefix code that encodes the string using the fewest possible bits, among all possible binary prefix codes for these symbols. This algorithm, known as Huffman coding.

Note that this algorithm assumes that we already know how many times each symbol occurs in the string so that we can compute the frequency of each symbol of dividing the number of times this symbol occurs by the length of the string.

Huffman coding is a fundamental aglorithm in data compression the subject devoted to reducing the number of bits required to represent information.

Huffman coding is extensively used to compress bit strings representing text and it also plays an important role in compressing audio and image files.

Algorithm : Huffman coding

Procedure Huffman (C : symbols a_i with frequencies W_i, $i = 1, ..., n$)

F : = forest of n rooted trees, each consisting of the single vertex a_i and assigned weight W_i.

While F is not a tree

begin

Replace the rooted trees T and T′ of least weights from F with $W(T) \geq W(T')$ with a tree having a new root that has T as its left subtree and T′ as its right subtree. Label the new edge to T with 0 and the new edge to T′ with 1.

Assign $W(T) + W(T')$ as the weight of the new tree.

end

{the Huffman coding for the symbol a_i is the concatenation of the labels of the edges in the unique path from the root to the vertex a_i}

Given symbols and their frequencies, out goal is to construct a rooted binary tree where the symbols are the labels of the leaves. The algorithm begins with a forest of trees each consisting of one vertex, where each vertex has a symbol as its label and where the weight of this vertex equals the frequency of the symbol that is its label. At each step we combine two trees having the least total weight into a single tree by introducing a new root and placing the tree with larger weight as its left subtree and the tree with smaller weight as its right subtree.

Furthermore, we assign the sum of the weights of the two subtrees of this tree as the total weight of the tree. The algorithm is finished when it has constructed a tree, that is, when the forest is reduced to a single tree.

Note that Huffman coding is a greedy algorithm. Replacing the two subtrees with the smallest weight at each step leads to an optimal code in the sense that no binary prefix code for these symbols can encode these symbols using fewer bits.

There are many variations of Huffman coding

For example, instead of encoding single symbols, we can encode blocks of symbols of a specified length, such as blocks of two symbols. Doing so many reduce the number of bits required to encode the string. We can also use more than two symbols to encode the original symbols in the string. Furthermore, a variation known as adaptive Huffman coding can be used when the frequency of each symbol in a string is not known in advance, so that encoding is done at the same time the string is being read.

In other words, a prefix code where in the shorter sequences are used for the more frequently occurring symbols. If there are many symbols, such as all 26 letters of the alphabet, a trial-and-error methods for constructing such a tree is not specified. An elegant construction developed by Huffman provides a technique for constructing such trees. The general problem of constructing an efficient tree can be described as follows :

Let $w_1, w_2, ... w_n$ be a set of positive numbers called weights, where $w_1 \le w_2 \le ...... \le w_n$.

If T = (V, E) is a complete binary tree with n leaves, assign these weights to the n leaves.

The result is called a complete binary tree for the weights $w_1, w_2, ..., w_n$.

The weight of the tree, denoted W(T), is defined as $\sum_{i=1}^{n} w_i l\,(w_i)$, where for each $1 \le i \le n$, $l(w_i)$ is the level number of the leaf assigned the weight w_i. The objective is to assign the weights so that W(T) is as small as possible.

A complete binary tree T′ for these weights is said to be an optimal tree if W(T′) ≤ W(T) for any other complete binary tree T for the weights.

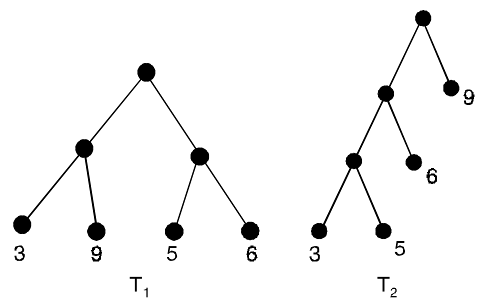


Fig. 3.75.

Fig. (3.75) shows two complete binary trees for the weights 3, 5, 6 and 9.

For tree T_1, $W(T_1) = \sum_{i=1}^{4} w_i l(w_i) = (3 + 9 + 5 + 6) . 2 = 46$

because each leaf has level number 2.

In the case of T_2, $W(T_2) = 3.3 + 5.3 + 6.2 + 9.1 = 45$, which is optimal.

Huffman construction is that in order to obtain an optimal tree T for the n weights $w_1, w_2, w_3, ..., w_n$, one considers an optimal tree T′ for the $n - 1$ weights $w_1 + w_2, w_3, ..., w_n$.

In particular, the tree T′ is transformed into T by replacing the leaf v having weight $w_1 + w_2$ by a tree rooted at v of height 1 with left child of weight w_1 and right child of weight w_2.

If the tree T_2 in Fig. (3.75) is optimal for the four weights 1 + 2, 5, 6, 9, then the tree in Fig. (3.76) will be optimal for the five weights 1, 2, 5, 6, 9.

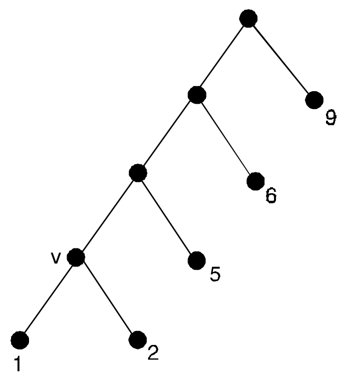


Fig. 3.76.

Example : For the prefix code P = {111, 0, 1100, 1101, 10} the longest binary sequence has length 4.

The labeled full binary tree of height 4 as shown in Fig. (3.77).

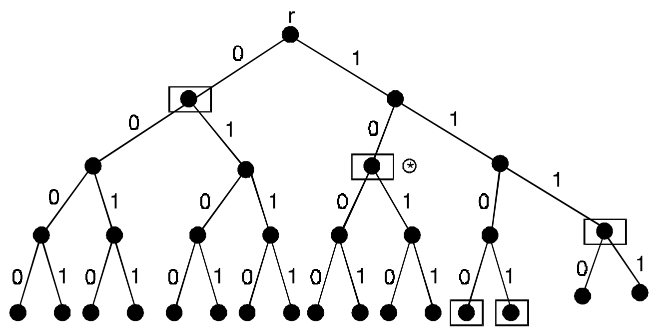


Fig. 3.77.

The elements of P are assigned to the vertices of this tree as follows.

For example, the sequence 10 traces the path from the root r to its right child C_R. Then it continues to the left child of C_R where the box indicates completion of the sequence.

Returning to the root, the other four sequences are traced out in similar lashion, resulting in the other four boxed vertices. For each boxed vertex remove the subtree that it determines.

The resulting tree is the complete binary tree of Fig. (3.78) where no box is an ancestor of another box.

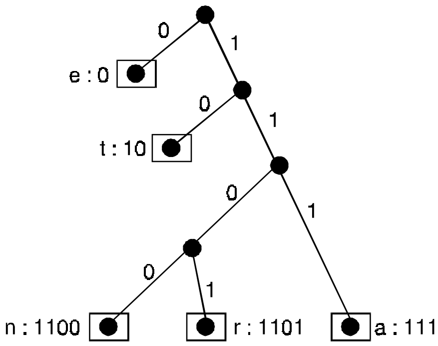


Fig. 3.78.

Lemma :

If T is an optimal tree for the n weights $w_1 \leq w_2 \leq .... \leq w_n$, then there exists an optimal tree T′ in which the leaves of weights w_1 and w_2 are siblings at the maximal level (in T′).

Proof. Let v be an internal vertex of T where the level number of v is maximal for all internal vertices. Let w_x and w_y be the weights assigned to the children x, y of vertex v, with $w_x \leq w_y$.

By the choice of vertex v, $l(w_x) = l(w_y) \geq l(w_1), l(w_2)$.

Consider the case of $w_1 < w_x$. (If $w_1 = w_x$, then w_1 and w_x can be interchanged and we would consider the case of $w_2 < w_y$. Applying the following proof to this case, we would find that w_y and w_2 can be interchanged).

If $l(w_x) > l(w_1)$, let $l(w_x) = l(w_1) + j$, for some $j \in Z^+$. Then $w_1 l(w_1) + w_x l(w_x) = w_1 l(w_1) + w_x\,[l(w_1) + j]$

$$= w_1 l(w_1) + w_x j + w_x l(w_1)$$

$$> w_1 l(w_1) + w_1 j + w_x l(w_1)$$

$$= w_1 l(w_x) + w_x l(w_1).$$

So $\quad W(T) = w_1 l(w_1) + w_x l\,(w_x) + \sum_{i \neq 1,\, x} w_i l(w_i) > w_1 l(w_x) + w_x l(w_1) + \sum_{i \neq 1,\, x} w_i l(w_i)$.

Consequently, by interchanging the locations of the weights w_1 and w_x, we obtain a tree of smaller weight.

But this contradicts the choice of T as an optimal tree.

Therefore, $l(w_x) = l(w_1) = l(w_y)$.

In a similar manner, it can be shown that

$$l(w_y) = l(w_2), \text{ so } l(w_x) = l(w_y) = l(w_1) = l(w_2).$$

Interchanging the locations of the pair w_1, w_x and the pair w_2, w_y, we obtain an optimal tree T′, where w_1, w_2 are siblings.

Theorem 3.34. *Let T be an optimal tree for the weights $w_1 + w_2, w_3, ..., w_n$, where $w_1 \le w_2 \le w_3 ...... \le w_n$. At the leaf with weight $w_1 + w_2$ place a (complete) binary tree of height 1 and assign the weights w_1, w_2 to the children (leaves) of this former leaf. The new binary tree T_1 so constructed is then optimal for the weights $w_1, w_2, w_3, ... w_n$.*

Proof. Let T_2 be an optimal tree for the weights $w_1, w_2, ..., w_n$, where the leaves for weights w_1, w_2 are siblings.

Remove the leaves of weights w_1, w_2 and assign the weight $w_1 + w_2$ to their parent (now a leaf).

This complete binary tree is denoted T_3 and

$$W(T_2) = W(T_3) + w_1 + w_2. \text{ Also } W(T_1) = W(T_2) + w_1 + w_2.$$

Since T is optimal, $W(T) \le W(T_3)$.

If $W(T) < W(T_3)$, then $W(T_1) < W(T_2)$, contradicting the choice of T_2 as optimal.

Hence $W(T) = W(T_3)$ and, consequently, $W(T_1) = W(T)$.

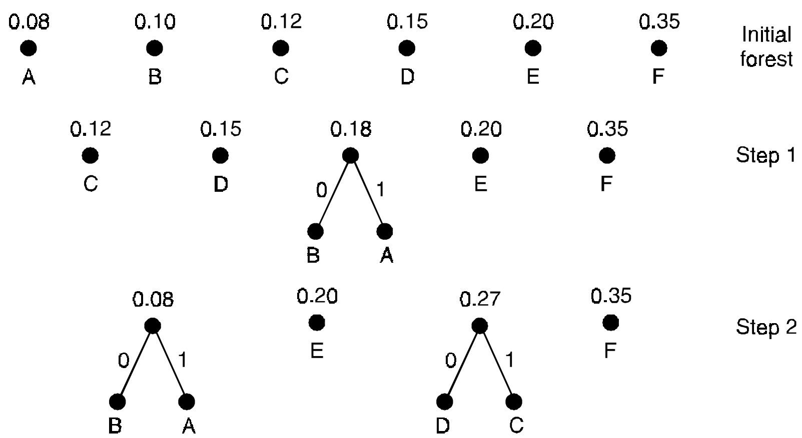
So T_1 is optimal for the weights $w_1, w_2, ..., w_n$.

Problem 3.68. *Use Huffman coding to encode the following symbols with the frequencies listed : A : 0.08, B : 0.10, C : 0.12, D : 0.15, E : 0.20, F : 0.35. What is the average number of bits used to encode a character ?*

Solution. Fig. (3.79) displays the steps used to encode these symbols. The encoding produced encodes A by 111, B by 110, C by 011, D by 010, and F by 00.

The average number of bits used to encode a symbol using this encoding is

$$3.0.08 + 3.0.10 + 3.0.12 + 3.0.15 + 2.0.20 + 2.0.35 = 2.45.$$



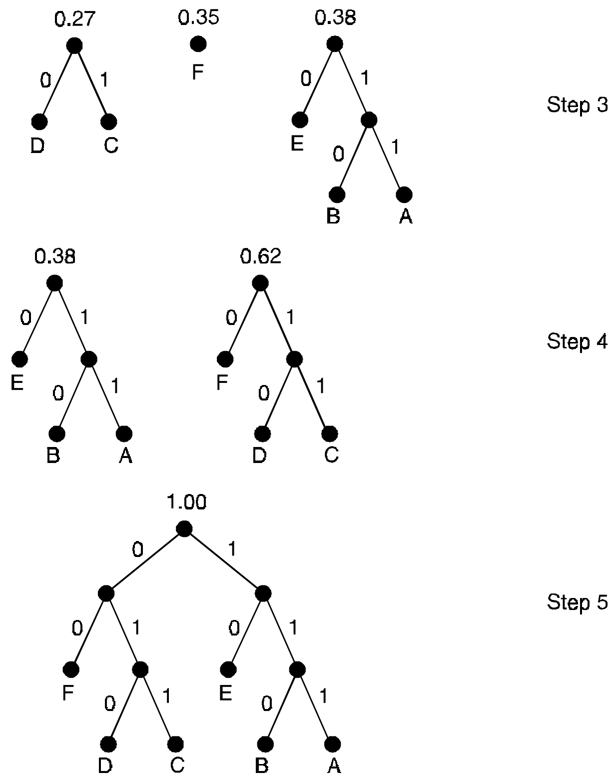


Fig. 3.79. Huffman coding of symbols.

Problem 3.69. *Construct two different Huffman codes for these symbols and frequencies : t : 0.2, u : 0.3, v : 0.2, w : 0.3.*

Solution. There are four possible answer in all, the one shown here and three more obtained from this one by swapping *t* and *v* and/or swapping *u* and *w*.

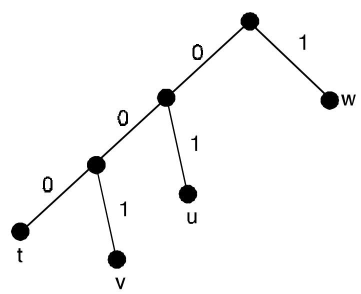


Fig. 3.80.

Problem 3.70. *Build a binary search tree for the words banana, peach, apple, pear, coconut, mango, and papaya using alphabetical order.*

Solution.

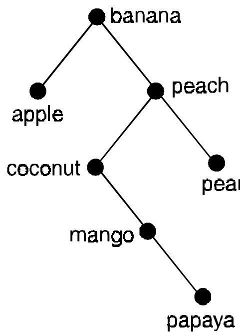


Fig. 3.81.

Problem 3.71. *Construct an optimal prefix code for the symbols a, 0, q, u, y, z that occur (in a given sample) with frequencies 20, 28, 4, 17, 12, 7, respectively.*

Solution. Fig. 3.82 shows the construction that follows Huffman's procedure. In part (*b*) weights 4 and 7 are combined so that we then consider the construction for the weights 11, 12, 17, 20, 28.

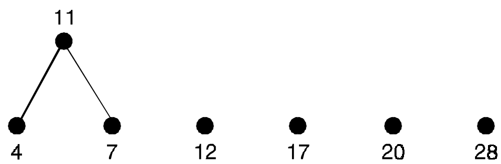
At each step (in part (*c*) – (*f*) of Fig. (3.82), we create a tree with subtrees rooted at the two smallest weights.

These two smallest weights belong to vertices each of which is originally either isolated (a tree with just a root) or the root of a tree obtained earlier in the construction. From the last result, a prefix code is determined as

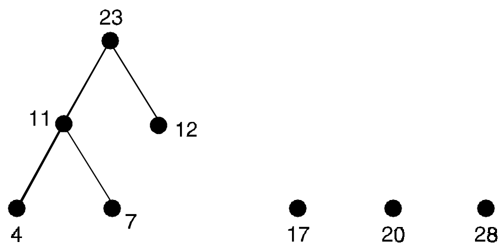
$$a : 11, \quad 0 : 01, \quad q : 0000, \quad u : 10, \quad y : 001, \quad z : 0001$$

(*a*)

(*b*)



(*c*)



(*d*)

(*e*)

(*f*)

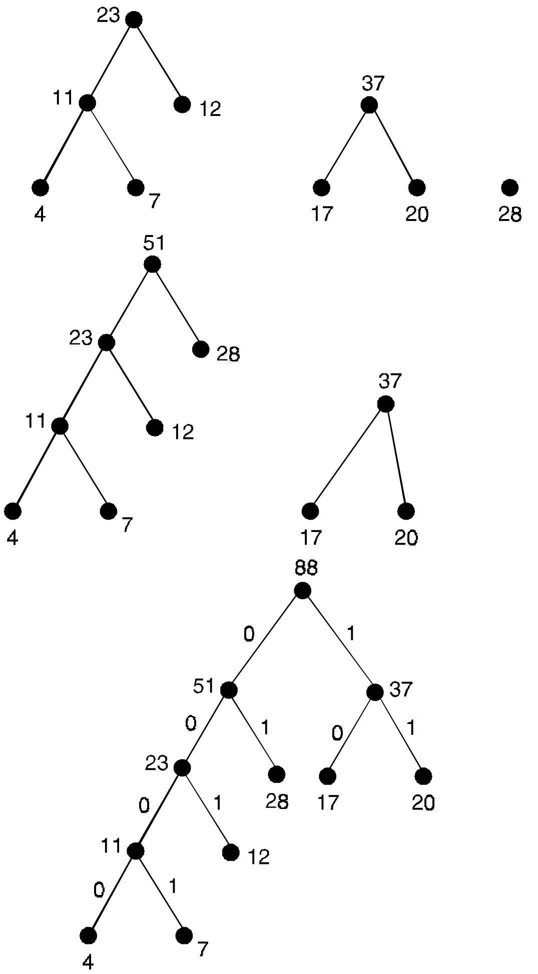


Fig. 3.82.

Problem 3.72. *For the prefix code given in Fig. (3.83), decode the sequences*

(*a*) *1001111101* (*b*) *10111100110001101* (*c*) *1101111110010*

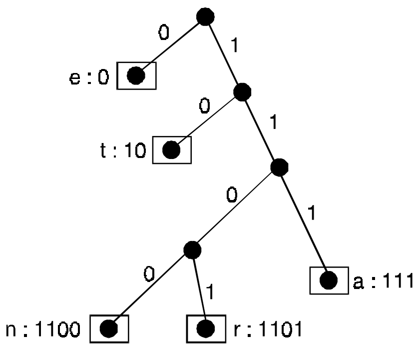


Fig. 3.83.

Solution. (*a*) *t e a r*

(*b*) *t a t e n e r*

(*c*) *r a n t.*

3.14. MORE APPLICATIONS

In this section we consider, graph theory applications, taken from operational research, organic chemistry, electrical network theory and computing and each involving the use of trees.

3.14.1. The minimum connector problem

Let us suppose that we wish to build a railway network connecting n given cities so that a passenger can travel from any city to any other. If, for economic reasons, the total amount of track must be a minimum, then the graph formed by taking the n cities as vertices and the connecting rails as edges must be a tree.

The problem is to find an efficient algorithm for deciding which of the n^{n-2} possible trees connecting these cities uses the least amount of track, assuming that the distances between all the pairs of cities are known.

As before, we can reformulate the problem in terms of weighted graphs. We denote the weight of the edge e by W(e), and our aim is to find the spanning tree T with least possible total weight W(T).

Unlike some of the problems we considered earlier, there is a simple algorithm that provides the solution. It is known as a greedy algorithm, and involves choosing edges of minimum weight in such a way that nocycle is created.

For example, if there are five cities, as shown in Fig. (3.84), then we start by choosing the edges AB (weight 2) and BD (weight 3).

We cannot then choose the edge AD (weight 4), since it would create the cycle ABD, so we choose the edge DE (weight 5). We cannot then choose the edges AE or BE (weight 6), since each would create a cycle, so we choose the edge BC (weight 7).

This completes the tree (see Fig. 3.85).

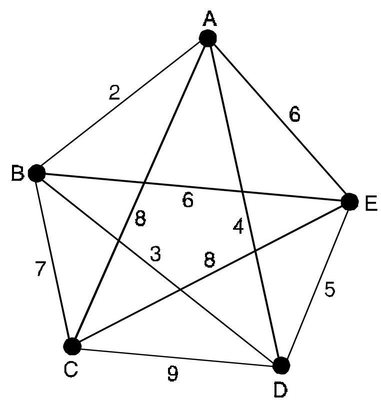


Fig. 3.84.

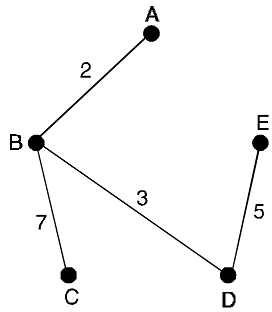


Fig. 3.85.

3.14.2. Enumeration of chemical molecules

One of the earliest uses of trees was in the enumeration of chemical molecules. If we have a molecule consisting only of carbon atoms and hydrogen atoms, then we can represent it as a graph in which each carbon atom appears as a vertex of degree 4, and each hydrogen atom appears as a vertex of degree 1.

The graphs of *n*-butane and 2-methyl propane are shown in Fig. (3.86). Although they have the same chemical formula C_4H_{10}, they are different molecules because the atoms are arranged differently with in the molecules. These two molecules form part of a general class of molecules known as the alkanes or paraffins, with chemical formula C_nH_{2n+2}, and it is natural to ask how many different molecules there are with this formula.

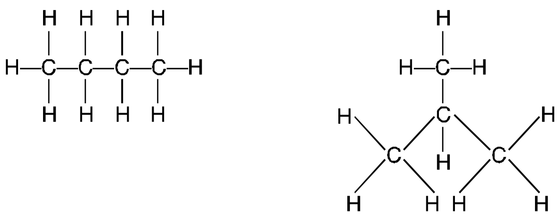


Fig. 3.86.

To answer this, we note first that the graph of any molecule with formula $C_n\ H_{2n+2}$ is a tree. Since it is connected and has $n + (2n + 2) = 3n + 2$ vertices and $\{4n + (2n + 2)\}/2 = 3n + 1$ edges.

Note that the molecule is determined completely once we know how the carbon atoms are arranged, since hydrogen atoms can then be added in such a way as to bring the degree of each carbon vertex to 4. We can thus discard the hydrogen atoms, as in Fig. (3.87), and the problem reduces to that of finding the number of trees with *n* vertices, each of degree 4 or less.

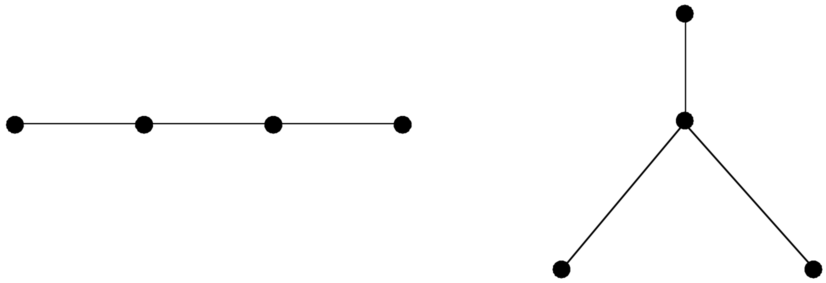


Fig. 3.87.

3.14.3. Electrical neworks

Suppose that we are given the electrical network in Fig. (3.88), and that we wish to find the current in each wire.

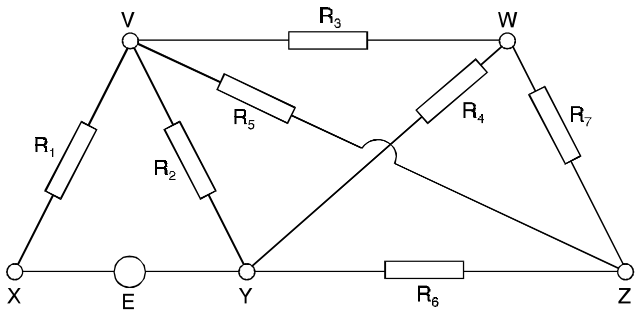


Fig. 3.88.

To do this, we assign an arbitrary direction to the current in each wire, as in Fig. (3.89), and apply Kirchhoff's laws :

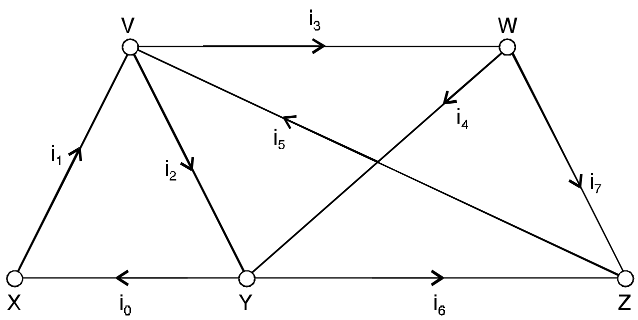


Fig. 3.89.

(*i*) the algebraic sum of the currents at each vertex is 0,

(*ii*) the toal voltage in each cycle is obtained by adding the products of the currents i_k and resistances R_k in that cycle.

Applying Kirchhoff's second law to the cycles VYXV, VWYV and VWYXV, we obtain the equations.

$i_1R_1 + i_2R_2 = E$, $i_3R_3 + i_4R_4 - i_2R_2 = 0$, $i_1R_1 + i_3R_3 + i_4R_4 = E$.

The last of these three equations is simply the sum of the first two, and gives us no further information. Similarly, if we have the Kirchhoff equations for the cycles VWYV and WZYW, then we can deduce the equation for the cycle VWZYV. It will save a lot of work if we can find a set of cycles that gives us the information we need without any redundancy, and this can be done by using a fundamental set of cycles.

In this example, taking the fundamental system of cycles in Fig. (3.90), we obtain the following equations.

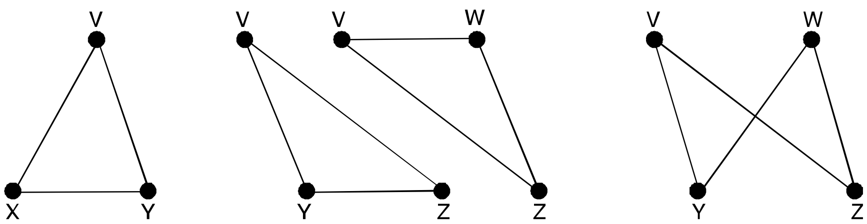


Fig. 3.90.

For the cycle VYXV, $i_1R_1 + i_2R_2 = E$,

For the cycle VYZV, $i_2R_2 + i_5R_5 + i_6R_6 = 0$,

For the cycle VWZV, $i_3R_3 + i_5R_5 + i_7R_7 = 0$,

For the cycle VYWZV, $i_2R_2 - i_4R_4 + i_5R_5 + i_7R_7 = 0$,

The equations arising from Kirchhoff's first law are :

For the vertex X, $i_0 - i_1 = 0$,

For the vertex V, $i_1 - i_2 - i_3 + i_5 = 0$,

For the vertex W, $i_3 - i_4 - i_7 = 0$

For the vertex Z, $i_5 - i_6 - i_7 = 0$

These eight equations can now be solved to give the eight currents i_0, ... i_7.

For example, if E = 12, and if each wire has unit resistance (that is, $R_i = 1$ for each i), then the solution is as given in Fig. (3.91).

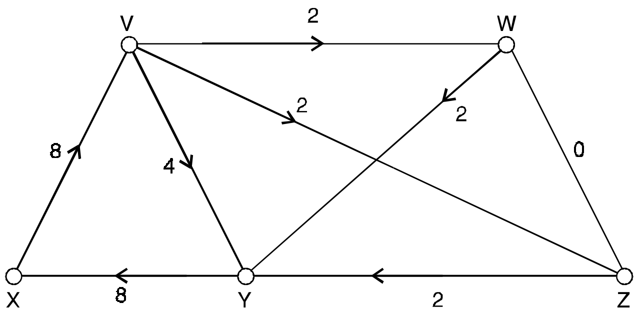


Fig. 3.91.

Problem Set 3.1

1. (*a*) Draw the graphs of all nonisomorphic unlabeled trees with five vertices.

(*b*) How many isomers does pentane (C_5H_{12}) have ? Why ?

2. Prove that the subgraph of a C_kH_{2k+2} tree T consisting of the k carbon vertices and all edges from T among them is itself a tree.

3. Recall that a graph is acyclic if it has no cycles. Prove that a graph with n vertices is a tree if and only if it is acyclic with $n - 1$ edges.

4. Prove that a connected graph with n vertices is a tree if and only if the sum of the degrees of the vertices is $2(n - 1)$

5. (*a*) Draw the graphs of all nonisomorphic unlabeled trees with six vertices.

(*b*) How many isomers does hexane (C_6H_{14}) have ? Why ?

6. Suppose G is an acyclic graph with $n \geq 2$ vertices and we remove one edge. Explain why the new graph G′ cannot be connected.

7. Suppose a graph G has two connected components, T_1, T_2, each of which is a tree. Suppose we add a new edge to G by joining a vertex of T_1 to a vertex in T_2. Prove that the new graph is a tree.

8. Let T be a tree with n vertices $v_1, v_2, \ldots\ldots v_n$. Prove that the number of leaves in T is

$$2 + \sum_{\deg v_i \geq 3} [\deg v_i - 2].$$

9. Prove that a tree with $n \geq 2$ vertices is a bipartite graph.

10. (*a*) Show that a tree with two vertices of degree 3 must have at least four vertices of degree 1.

(*b*) Show that the result of (*a*) is best possible. A tree with two vertices of degree 3 need not have five vertices of degree 1.

11. Suppose T is a tree with k vertices labeled C, each of degree at most 4. Enlarge T by adjoining sufficient vertices labeled H so that each vertex C has degree 4 and each vertex H has degree 1. Prove that the number of H vertices adjoined to the graph must be $2k + 2$.

12. Let e be an edge in a tree T. Prove that the graph consisting of all the vertices of T but with the single edge e deleted is not connected.

13. Let e be an edge of the complete graph k_n. Prove that the number of spanning trees of k_n which contain e is $2n^{n-3}$.

14. Suppose some edge of a connected graph G belongs to every spanning tree of G. What can you conclude and why ?

15. How many labeled trees are there on n vertices, for $1 \leq n \leq 6$?

16. How many spanning trees does k_7 have ? Why ?

17. Draw all the labeled trees on four vertices ?

18. Draw all the spanning trees of $K_{2,2}$ and indicate the isomorphism classes of these. How many isomorphism classes are there ?

19. Determine the number of spanning trees of the complete bipartite graph $K_{2,n}$.

20. How many graphs have n vertices labeled $v_1, v_2, \ldots, v_n$ and $n - 1$ edges ? Compare this number with the number of trees with vertices $v_1, \ldots, v_n$ for $2 \le n \le 6$.

21. (*a*) Prove that every edge in a connected graph is part of some spanning tree.

(*b*) Prove that any two edges of a connected graph are part of some spanning tree.

(*c*) Given three edges in a connected graph, is there always a spanning tree containing these edges ? Explain your answer.

22. If G is a graph and e is an edge which is not part of a circuit, then e must belong to every spanning tree of G. Why ?

23. Let C_n be the cycle with n vertices labeled 1, 2, ... n, in the order encountered on the cycle.

(*a*) Find the number of spanning trees for C_n (in two ways).

(*b*) Find a general formula for the number of spanning trees for $C_n \cup \{e\}$ where e joins 1 to a $(3 \le a \le n - 1)$.

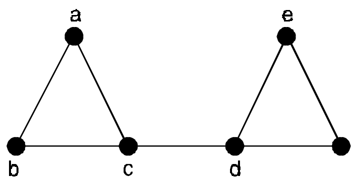
24. Draw three distinct rooted trees that have 4 vertices.

25. Find all the trees with six vertices.

26. Define spanning tree. Give an example.

27. Which connected simple graphs have exactly one spanning tree ?

28. Draw all the spanning trees of the following graph shown below.

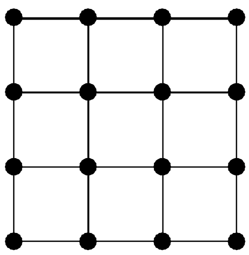


29. How many different spanning trees does each of the following simple graphs have ?

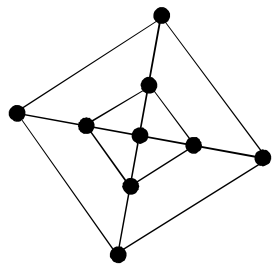
(*a*) K_3 (*b*) K_4 (*c*) $K_{2, 2}$.

30. Find a spanning tree for each of the graphs shown by removing edges.

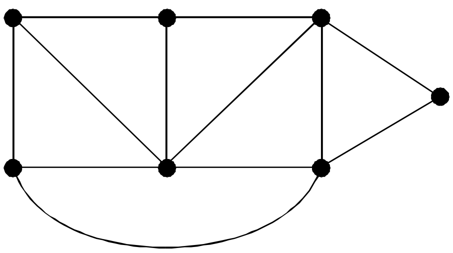
(*a*)



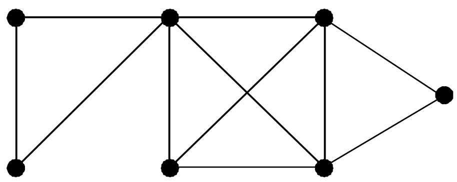
(*b*)



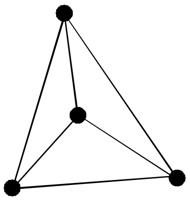
(*c*)



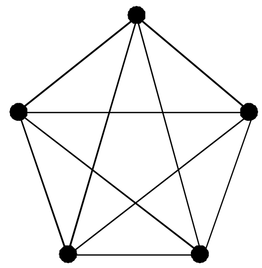
(*d*)



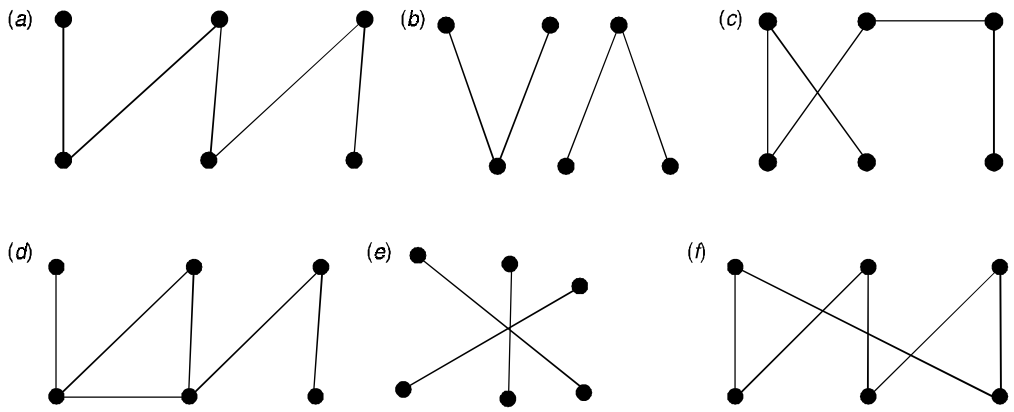
(*e*)



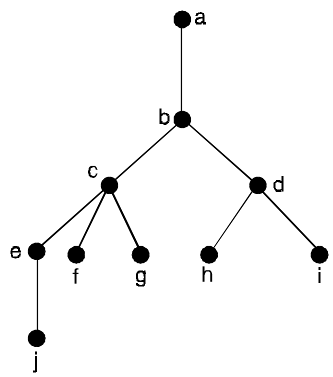
(*f*)



31. Which of the following graphs are tree ?

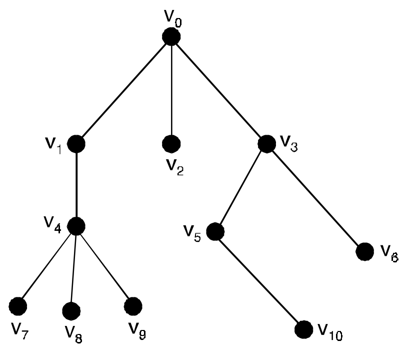


32. Give the tree with root at a as shown in Figure.



(*a*) find the parents of *c* and *h*
(*b*) find the children of *d* and *e*
(*c*) find the descendents of *c* and *e*
(*d*) find the siblings of *f* and *h*
(*e*) find the leaves
(*f*) find the internal vertices
(*g*) draw the subtree rooted at *c*.

33. Consider the tree with root v_0 shown in Figure.



(*a*) what are the levels of v_0 and v_4 ?

(*b*) what are the children of v_3 ?

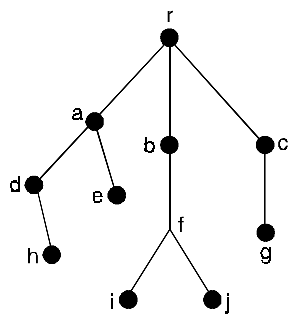
(*c*) what is the height of this rooted tree ?

(*d*) what is the parent of v_5 ?

(*e*) what are the siblings of v_7 ?

(*f*) what are the descendants of v_3 ?

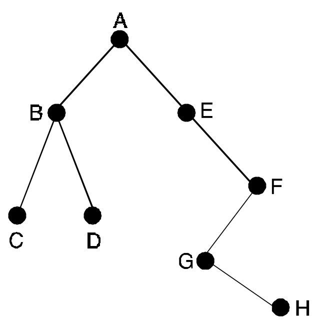
34. Consider the rooted tree T in Figure



(*a*) identifying path from the root *r* to each of the following vertices, and find the level of vertex (*i*) *d* (*ii*) *j* and (*iii*) *g*.

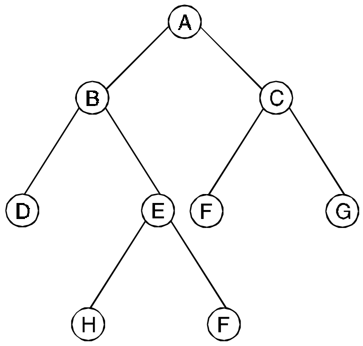
(*b*) find the leaves of T

35. Give linked representation of the binary tree

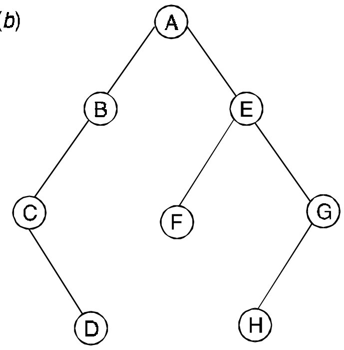


36. Give an array representation of each tree given below

(*a*)

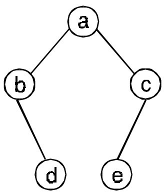


(*b*)

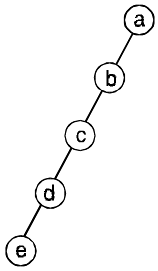


37. Determine the order in which the vertices of the binary tree given below will be visited under
(*i*) In-order (*ii*) pre-order (*iii*) post-order.

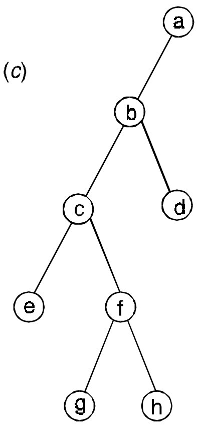
(*a*)



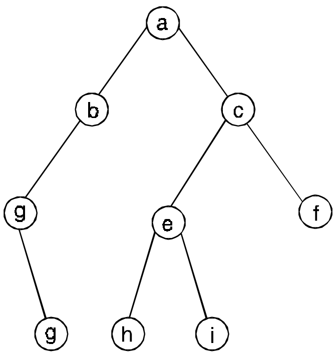
(*b*)



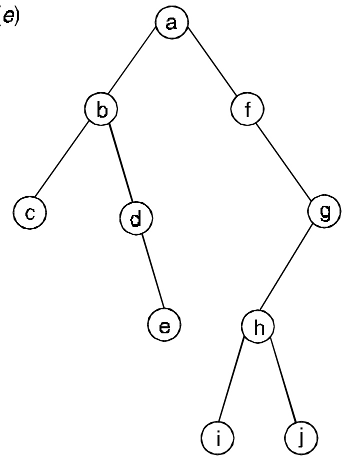
(*c*)



(*d*)



(*e*)



38. How many binary trees are possible with three vertices ?

39. How will you different between a general tree and a binary tree ?

40. Draw two different binary trees with five vertices having maximum number of leaves.

41. Construct a binary tree whose in-order and pre-order traversal is given below

(*a*) (*i*) In-order : 5, 1, 3, 11, 6, 8, 2, 4, 7

(*ii*) Pre-order : 6, 1, 5, 11, 3, 4, 8, 7, 2

(*b*) (*i*) In-order : 10, 8, 9, 7, 6, 4, 5, 2, 3, 1

(*ii*) Pre-order : 10, 9, 8, 7, 6, 5, 4, 3, 2, 1

(*c*) (*i*) In-order : *d*, *g*, *b*, *e*, *i*, *h*, *j*, *a*, *c*, *f*

(*ii*) Pre-order : *a*, *b*, *d*, *g*, *e*, *h*, *i*, *j*, *c*, *f*.

42. Draw the binary tree to represent the expression $(x + 3y)^5 (a - 2b)$ and find the expression in pre-order notation.

43. Draw the binary tree to represent the expression $(((a * b) - c) \uparrow d) - ((e * f) + g)$

Write the corresponding expression in post order notations and hence find the value when $a = 3$, $b = 4$, $c = 5$, $d = 2$, $e = 6$, $f = 7$ and $g = 1$.

44. Represent each of the expression in a binary tree

(*a*) (A + B) * (C – D)

(*b*) [(A + B)/C] + D

(*c*) [(A – B) ↑ 2]/(A + B)

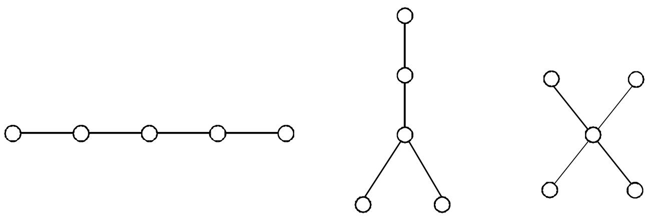
(*d*) (X + 7) * [(4 * Y + Z)/(S + 3)]

(*e*) $((3 + x) - (4 * x)) - (x - 2)$

(*f*) $((5 * a) + (3 - (6 * a)) + (a - 3 * b))$.

Answers 3.1

1. (*a*)



(*b*) Since each tree with five vertices has all vertices of degree atmost four, there is one isomer for each such tree, the C atoms corresponding to the vertices. There are three isomers of C_5H_{12}.

2. There are no circuits in the subgraph since there are no circuits in C_kH_{2k+2}. Also, given any two C vertices, there is a path between them in C_kH_{2k+2} (because C_kH_{2k+2} is connected). Any H vertex on this path would have degree two. Thus, there is none, the path consists entirely of C vertices and hence lies with in the subgraph. Thus, the subgraph is connected hence a tree.

3.(→) A tree with n vertices has $n - 1$ edges and no cycles.

(←) Suppose G is an acyclic graph with n vertices and $n - 1$ edges. Since G has no cycles, we have only to prove that G is connected. Let then $C_1, C_2, \ldots, C_k$ be the connected components of G and suppose that C_i has n_i vertices. (Thus, $\Sigma n_i = n$). Since G has no cycles, there are no cycles with in each C_0. It follows that each C_i is a tree with $n_i - 1$ edges. The number of edges in G is, therefore $\sum_{1}^{k}(n_i - 1) = \left(\sum_{1}^{k} n_i\right) - k = n - k$, So $n - k = n - 1$, $k = 1$, G has only one component i that is , G is connected.

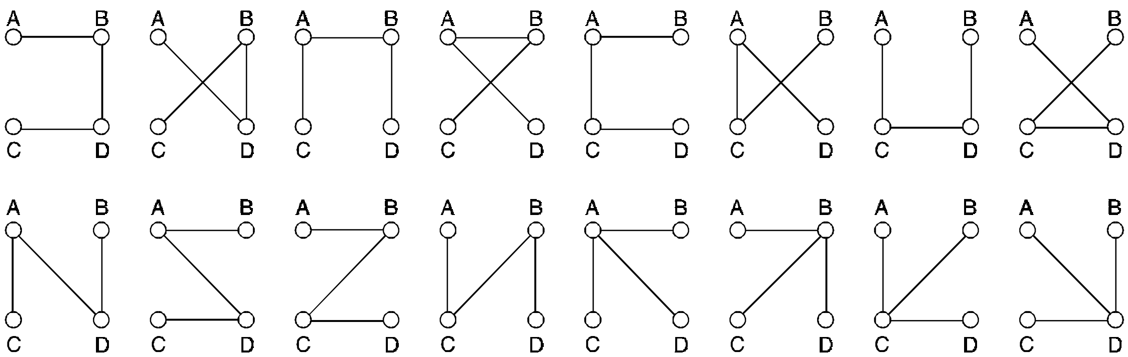
10. (*a*) We have $\Sigma\deg(v_i) \geq 8$, so the tree has at least four edges and hence at least five vertices. If the result is not true, then there are at most three vertices of degree one while the rest have degree at least two. Then $\Sigma\deg v_i \geq 2(3) + 3(1) + (n - 5)2 = 2n - 1$ contradicting the fact that $\Sigma\deg v_i = 2(n - 1)$.

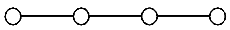
11. Let x be the number of H vertices adjoined. Since T had K – 1 edges, and one new edge is added for each H, G has $(\text{K} - 1) + x$ edges. Therefore $\Sigma\deg v_i = 2(\text{K} - 1 + x)$. But $\Sigma\deg v_i = 4\text{K} + x$ since each C has degree 4 and each H has degree 1. Therefore $4\text{K} + x = 2\text{K} - 2 + 2x$ and $x = 2\text{K} + 2$.

14. The edges in question is a bridge, that is, its removal disconnects the graph. To see why, call the edge e. If $G \backslash \{e\}$ were connected, it would have a spanning tree. However, since $G \backslash \{e\}$ contains all the vertices of G, any spanning tree for it is also a spanning tree for G. We have a contradiction.

15. The numbers $1^{-1} = 1$, $2^0 = 1$, $3^1 = 3$, $4^2 = 16$, $5^3 = 125$ and $6^4 = 1296$.

17.



18. $K_{2,2}$ has four spanning trees (obtained by deleting each edge in succession). They are all isomorphic to 

20. There are $\binom{n}{2}$ possible edges from which we choose $n - 1$. The number of graphs is, therefore

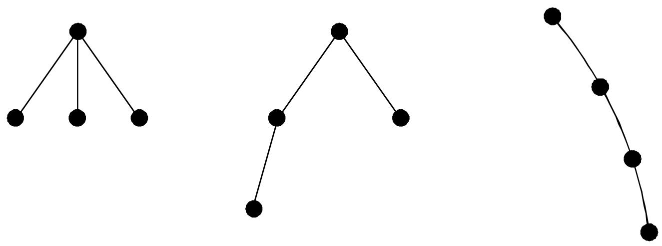
$\binom{\binom{n}{2}}{n-1}$. The number of trees on n labeled vertices is n^{n-2}. For $n \leq 6$, the table shows the numbers of trees Vs. graphs.

n	*No. of trees*	*No. of graphs*
2	1	1
3	3	3
4	16	20
5	125	210
6	1296	3003

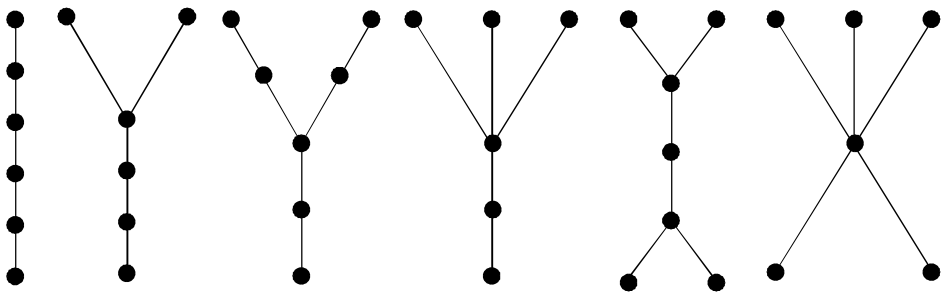
21. (*a*) Say the edge is e and T is any spanning tree. If e is not in T, then T $\cup$ $\{e\}$ must contain a circuit. Deleting and edge of this circuit other than e gives another spanning tree which includes e.

(*b*) No. If the three edges form a circuit, no spanning tree can contain them.

24.

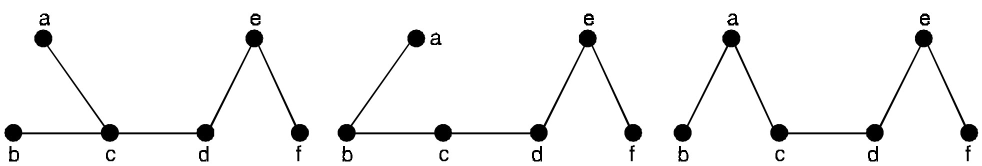


25.



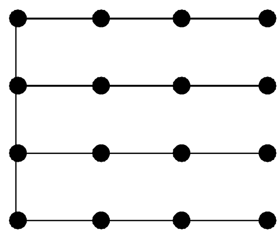
27. Tree.

28.

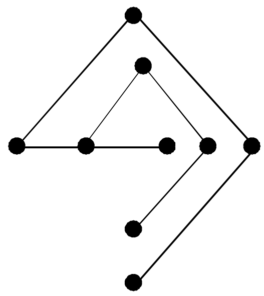


29. (*a*) 3 (*b*) 16 (*c*) 4

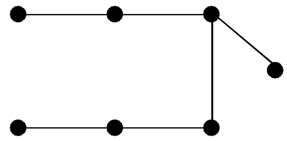
30. (*a*)



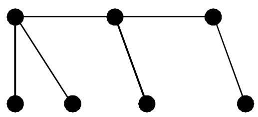
(*b*)



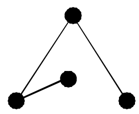
(*c*)



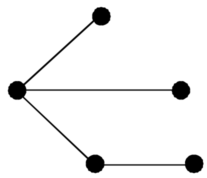
(*d*)



(*e*)



(*f*)

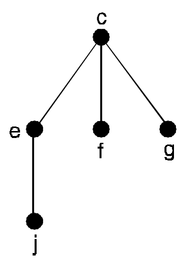


31. (*a*), (*c*), (*e*)

32. (*a*) *b* and *d* (*b*) for *d*, *h* and *i* and for *e* is *j* (*c*) *e*, *f*, *g*, *j* ; *j*

(*d*) *g* ; *i* (*e*) *j*, *f*, *g*, *h*, *i* (*f*) *a*, *b*, *c*, *d*, *e*

(*g*)

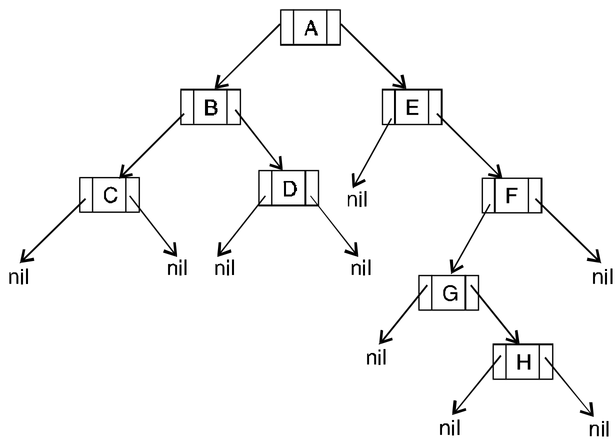


33. (*a*) 0, 2 (*b*) v_5, v_6 (*c*) 3 (*d*) v_3 (*e*) v_8, v_9 (*f*) v_5, v_6, v_{10}.

34. (*a*) (*i*) $r – a – d$; 2 (*ii*) $r – b – f – g$; 3 (*iii*) $r – c – g$; 2

(*b*) *h*, *e*, *i*, *j*, *g*

35.



36. (*a*)

1	2	3	4	5	6	7	8	9	10	11
A	B	C	D	E	F	G			H	I

(*b*)

1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	B	E	C		F	G		D					H

37. (*a*) *b d a e c*   *a b c d e*   *d b e c a*

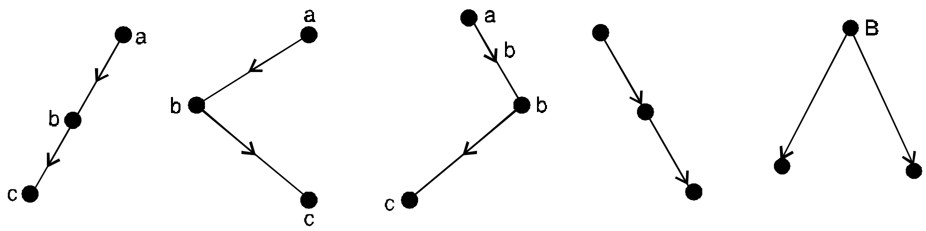
(*b*) *e d c b a*   *a b c d e*   *e d c b a*

(*c*) *e c g f h b d a*   *a b c e f g h d*   *e g h f c d b a*

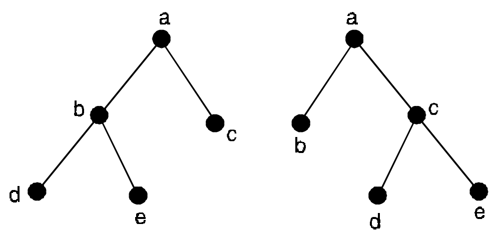
(*d*) *d g b a h e i c f*   *a b d g c e h i f*   *g d b h i e f c a*

(*e*) *c b d e a f i g h j g*   *a b c d e f g h i j*   *c e d b i j h g f a.*

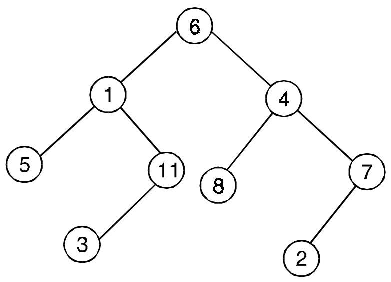
38. Five



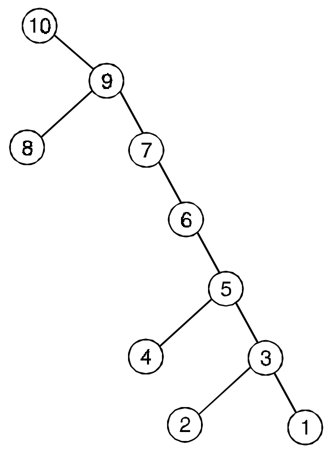
40.



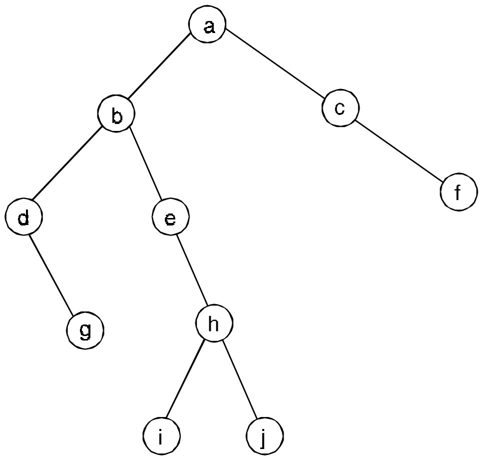
41. (*a*)



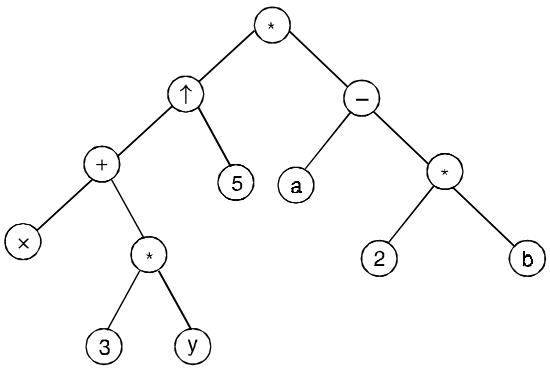
(*b*)



(*c*)

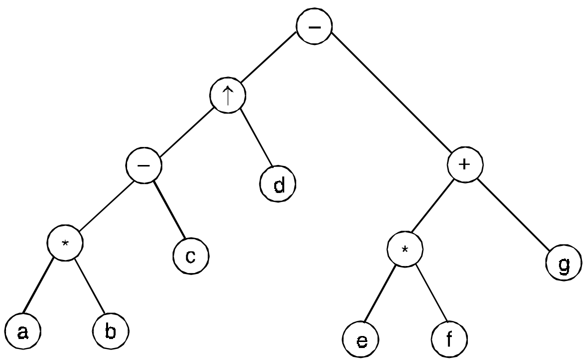


42.



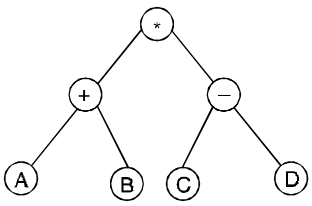
preorder $+ \uparrow + x * 3\, y\, 5 - a\, 2 + b$

43.

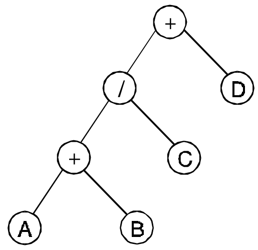


post order : $ab * c - d \uparrow ef\ * g + -$: 6.

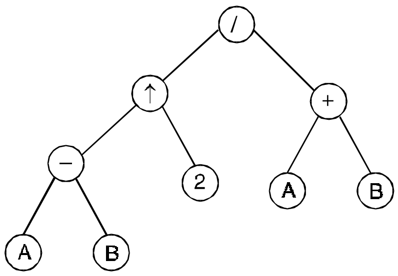
44. (*a*)



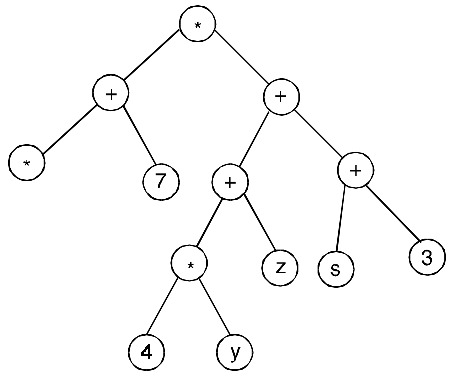
(*b*)



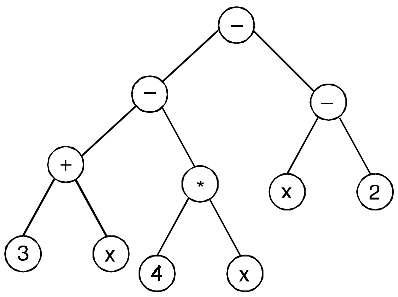
(*c*)



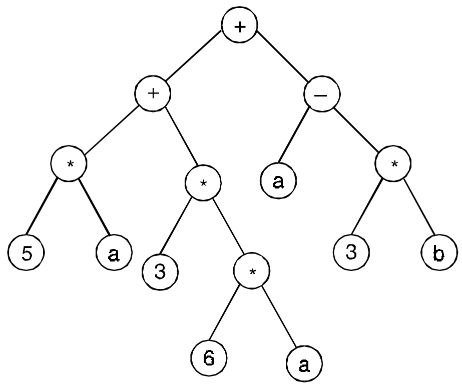
(*d*)



(*e*)



(*f*)



CHAPTER

4 Optimization and Matching

4.1 SHORTEST PATH ALGORITHMS

In many areas like transportation, cartoon motion planning, communication network topology design etc, problems related to finding shortest path algorithms.

The shortest path problem is concerned with finding the least cost (that costs minimum) path from an originating node in a weighted graph to a destination node in that graph. Let us consider a graph shown in the Fig. (4.1) in which number associated with each arc represent the weight of the arc.

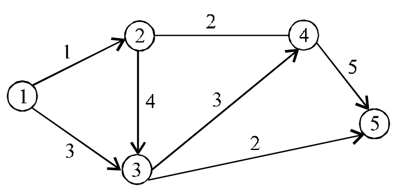


Fig. 4.1

There exist many algorithms for finding the shortest path in a weighted graph. One such algorithm is developed by Dijkstra in the early 1960's for finding the shortest path in a graph with non-negative weight associated with edge/arc without explicitly enumerating all possible paths. This algorithm is based upon a technique known as dynamic programming.

This algorithm determines shortest path between a pair of nodes in a graph. If there are n nodes in a graph, we need to run the algorithm ${}^{n}C_2$ times. In a network of 100 or more nodes, the time taken to compute the shortest path for all possible pair of nodes can be any body's guess.

To overcome this we shall discuss a modification of Dijkstra's alogrithm to find shortest distance between one node to all other nodes in a graph and Floyd Warshall's algorithm to compute all pair shortest path.

4.2 DIJKSTRA'S ALGORITHM

To find a shortest path from vertex A to vertex E in a weighted graph, carry out the following procedure.

Step 1 : Assign to A the label (–, 0).

Step 2 : Until E is labeled or no further labels can be assigned, do the following

(*i*) For each labeled vertex $u(x, d)$ and for each unlabeled vertex v adjacent to u, compute $d + w(e)$, where $e = uv$.

(*ii*) For each labeled vertex u and adjacent unlabeled vertex v giving minimum $d' = d + w(e)$, assign to v the label (u, d'). If a vertex can be labeled (x, d') for various vertices x, make any choice.

4.2.1. Dijkstra's algorithm (Improved)

To find the length of a shortest path from vertex A to vertex E in a weighted graph, proceed as follows:

Step 1 : Set v_1 = A and assign to this vertex the permanent label O.

Assign every other vertex a temporary label of ∞, where ∞ is a symbol which, by definition, is deemed to be larger than any real number.

Step 2 : Until E has been assigned a permanent label or no temporary labels are changed in (*a*) or (*b*), do the following.

(*a*) Take the vertex v_i which most recently acquired a permanent label, say *d*. For each vertex *v* which is adjacent to v_i and has not yet received a permanent label, if $d + w(v_i v) < t$ the current temporary label of *v* to $d + w(v_i v)$.

(*b*) Take a vertex *v* which has a temporary smallest among all temporary labels in the graph. Set $v_{i+1} = v$ and make its temporary label permanent. If there are several vertices *v* which tie for smallest temporary label, make any choice.

4.2.2. Floyd-Warshall algorithm

To find the shortest distances between all pairs of vertices in a weighted graph where the vertices are $v_1, v_2, ..., v_n$, carry out the following procedure:

Step 1 : For $i = 1$ to n, set $d(i, i) = 0$,

For $i \neq j$, if $v_i v_j$ is an edge, let $d(i, j)$ be the weight of this edge; otherwise

Set $d(i, j) = \infty$.

Step 2 : For $k = 1$ to n,

for $i, j = 1$ to n,

let $d(i, j) = \min \{d(i, j), d(i, k) + d(k, j)\}$

The final value of $d(i, j)$ is the shortest distance from v_i to v_j.

Problem 4.1. *Apply Dijkstra's algorithm to the graph given below and find the shortest path from a to f.*

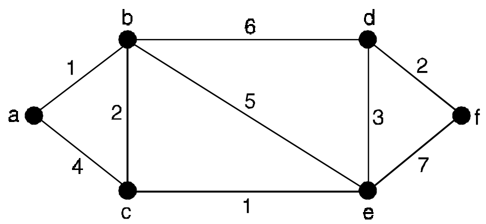


Fig. 4.2.

Solution. The initial labelling is given by

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

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

$$L(b) = \min\{\text{old } L(b), L(a) + w(ab)\}$$
$$= \min\{\alpha, 0 + 1.0\} = 1.0$$
$$L(c) = \min\{\text{old } L(c), L(a) + w(ac)\}$$
$$= \min\{\alpha, 0 + 4.0\} = 4.0$$

Hence minimum label is $L(b) = 1.0$

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*
L(*v*)	0	1.0	4.0	α	α	α
T	{	*b*,	*c*,	*d*,	*e*,	*f*}

Iteration 2 : $u = b$, the permanent label of b is 1.0 T becomes T–$\{b\}$ there are three edges incident with b, *i.e.*, bc, bd and be where $c, d, e \in$ T.

$$L(c) = \min\{\text{old } L(c), L(b) + w(bc)\}$$
$$= \min\{4.0, 1.0 + 2.0\} = 3.0$$
$$L(d) = \min\{\text{old } L(d), L(b) + w(bd)\}$$
$$= \min\{\alpha, 1.0 + 6.0\} = 7.0$$
$$L(e) = \min\{\text{old } L(e), L(b) + w(be)\}$$
$$= \min\{\alpha, 1.0 + 5.0\} = 6.0$$

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*
L(*v*)	0	1.0	3.0	7.0	6.0	α
T	{		*c*,	*d*,	*e*,	*f*}

Thus minimum label is $L(c) = 3.0$.

Iteration 3 : $u = c$, the permanent label of e is 3.0, T becomes T–$\{c\}$. There is one edge incident with c, *i.e.*, c, e where $e \in$ T.

$$L(c) = \min\{\text{old } L(e), L(c) + w(ce)\}$$
$$= \min\{6.0, 3.0 + 1.0\} = 4.0$$

Thus minimum label is $L(c) = 4.0$

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*
L(*v*)	0	1.0	3.0	7.0	4.0	α
T	{			*d*,	*e*,	*f*}

Iteration 4 : $u = e$, the permanent label of e is 4.0, T becomes T–$\{e\}$. There are two edges incident with e, *i.e.*, ed and ef where $d, f \in$ T

$$L(d) = \min \{\text{old } L(d), L(e) + w(ed)\}$$
$$= \min \{7.0, 4.0 + 3.0\} = 7.0$$
$$L(f) = \min \{\text{old } L(f), L(e) + w(ef)\}$$
$$= \min \{\alpha, 4.0 + 7.0\} = 11.0$$

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*
L(*v*)	0	1.0	3.0	7.0	4.0	11.0
T	{			*d*,		*f*}

Thus minimum label is L(*d*) = 7.0

Iteration 5 : *u* = *d*, the permanent label of *d* is 7.0. T becomes T–{*d*}. There is one edge incident with *d*, *i.e.*, *d*, *f* where $f \in T$

$$L(f) = \min \{\text{old } L(f), L(d) + w(df)\}$$
$$= \min \{11.0, 7.0 + 2.0\} = 9.0$$

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*
L(*v*)	0	1.0	3.0	7.0	4.0	9.0
T	{					*f*}

The minimum label is L(*f*) = 9.0

Since *u* = *f*, the only choice. Iteration stops. Thus the shortest distance between *a* and *f* is 9, and moreover, the shortest paths is {*a*, *b*, *c*, *e*, *d*, *f*}.

Problem 4.2. *Determine a shortest path between the vertices a to z as shown below.*

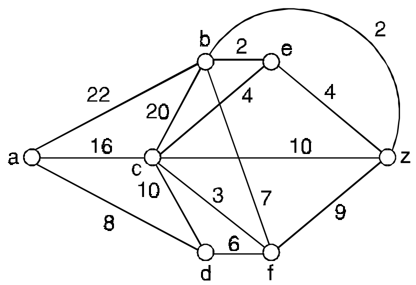


Fig. 4.3.

Solution. The initial labelling is given by

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

Iteration 1 : $u = a$ has $L(u) = 0$, T becomes T–{a}. There are three edges incident with a, *i.e.,* ab, ac, and ad where $b, c, d \in$ T

$$L(b) = \min \{\text{old } L(b), L(a) + w(ab)\}$$
$$= \min \{\alpha, 0 + 22\} = 22$$
$$L(c) = \min \{\text{old } L(c), L(a) + w(ae)\}$$
$$= \min \{\alpha, 0 + 16\} = 16$$
$$L(d) = \min \{\text{old } L(d), L(a) + w(ad)\}$$
$$= \min \{\alpha, 0 + 8\} = 8.$$

Hence minimum label is $L(d) = 8$.

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*	*z*
L(v)	0	22	16	8	α	α	α
T	{	*b*,	*c*,	*d*,	*e*,	*f*,	*z*}

Iteration 2 : $u = d$, the permanent label of d is 8. T becomes T–{d}. There are two edges incident with d, *i.e.,* dc and df where $c, f \in$ T.

$$L(c) = \min \{\text{old } L(c), L(d) + w(dc)\}$$
$$= \min \{16, 8 + 10\} = 16$$
$$L(f) = \min \{\text{old } L(f), L(d) + w(df)\}$$
$$= \min \{\alpha, 8 + 6\} = 14.$$

Hence minimum label is $L(f) = 14$.

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*	*z*
L(v)	0	22	16	8	α	14	α
T	{	*b*,	*c*,		*e*,	*f*,	*z*}

Iteration 3 : $u = f$, the permanent label of f is 14. T becomes T–{f}. There are three edges incident with f, *i.e.,* fc, fb and fz where $a, b, z \in$ T.

$$L(c) = \min \{\text{old } L(c), L(f) + w(fc)\}$$
$$= \min \{16 + 14 + 3\} = 16 \{16, + 14 + 3\}$$
$$L(b) = \min \{\text{old } L(b), L(f) + w(fb)\}$$
$$= \min \{22, 14 + 7\} = 21$$
$$L(z) = \min \{\text{old } L(z), L(f) + w(fz)\}$$
$$= \min \{\alpha, 14 + 9\} = 23.$$

Hence minimum label is $L(c) = 16$.

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*	*z*
L(v)	0	21	16	8	α	14	23
T	{	*b*,	*c*,		*e*,		*z*}

Iteration 4 : $u = c$, the permanent label of c is 16. T becomes T–{c}. These are three edges incident with c, *i.e.*, cb, ce and cz, where $b, e, z \in$ T.

$$L(b) = \min \{\text{old } L(b), L(c) + w(cb)\}$$
$$= \min \{21, 16 + 20\} = 21$$
$$L(e) = \min \{\text{old } L(e), L(c) + w(ce)\}$$
$$= \min \{\alpha, 16 + 4\} = 20$$
$$L(z) = \min \{\text{old } L(z), L(c) + w(cz)\}$$
$$= \min \{23, 16 + 10\} = 23.$$

Hence minimum label is L(e) = 20

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*	*z*
L(v)	0	21	16	8	20	14	23
T		*b*,			*e*		*z*

Iteration 5 : $u = e$, the permanent label of e is 20. T becomes T–{e}. There are two edges incident with e, *i.e.*, eb and ez where $b, z \in$ T.

$$L(b) = \min \{\text{old } L(b), L(e) + w(eb)\}$$
$$= \min \{21, 20 + 2\} = 21$$
$$L(z) = \min \{\text{old } L(z), L(e) + w(ez)\}$$
$$= \min \{23, 20 + 4\} = 23$$

Hence minimum label is L(b) = 21.

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*	*z*
L(v)	0	21	16	8	20	14	23
T		*b*,					*z*

Iteration 6 : $u = b$, the permanent label of b is 21. T becomes T–{b}. There is one edge incident with b. *i.e.*, bz where $z \in$ T.

$$L(z) = \min \{\text{old } L(z), L(b) + w(bz)\}$$
$$= \min \{23, 21 + 4\} = 23.$$

Hence minimum label is L(z) = 23.

Vertex V	*a*	*b*	*c*	*d*	*e*	*f*	*z*
L(v)	0	21	16	8	20	14	23
T	{						2}

Since $u = z$, the only choice iteration stops.

Thus the length of the shortest path is 23 and the shortest path is (a, d, f, z).

Problem 4.3. *What is the length of a shortest path between a and z in the weighted graph shown in Fig, (4.4)* ?

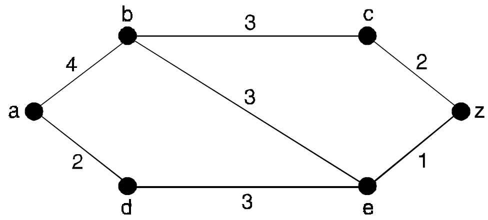


Fig. 4.4.

Solution. Although a shortest path is easily found by inspection, we will develop some ideas useful in understanding Dijkstra's algorithm.

We will solve this problem by finding the length of a shortest path from *a* to successive vertices, until *z* is reached.

The only paths starting at *a* that contain no vertex other than *a* (until the terminal vertex is reached) are *a*, *b* and *a*, *d*.

Since the lengths of *a*, *b* and *a*, *d* are 4 and 2, respectively, it follows that *d* is the closest vertex to *a*.

We can find the next closest vertex by looking at all paths that go through only *a* and *d* (until the terminal vertex is reached).

The shortest such path to *b* is still *a*, *b*, with length 4, and the shortest such path to *e* is *a*, *d*, *e* with length 5.

Consequently, the next closest vertex to *a* is *b*.

To find the third closest vertex to *a*, we need to examine only paths that go through only *a*, *d*, and *b* (until the terminal vertex is reached).

There is a path of length 7 to *c*, namely *a*, *b*, *c*, and *a* path of length 6 to *z*, namely, *a*, *d*, *e*, *z*.

Consequently, *z* is the next closest vertex to *a*, and the length of a shortest path to *z* is 6.

Problem 4.4. *Use Dijkstra's algorithm to find the length of a shortest path between the vertices a and z in the weighted graph displayed in Fig. (4.5).*

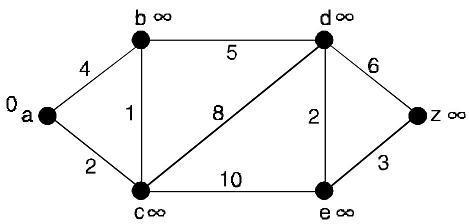


Fig. 4.5.

Solution. The steps used by Dijkstra's algorithm to find a shortest path between *a* and *z* are show in Fig. (4.6).

All each iteration of the algorithm the vertices of the set S_k are circled.

A shortest path from *a* to each vertex containing only vertices in S_k is indicated for each iteration.

The algorithm terminates when *z* is circled.

We find that a shortest path from *a* to *z* is *a*, *c*, *b*, *d*, *e*, *z*, with length 13.

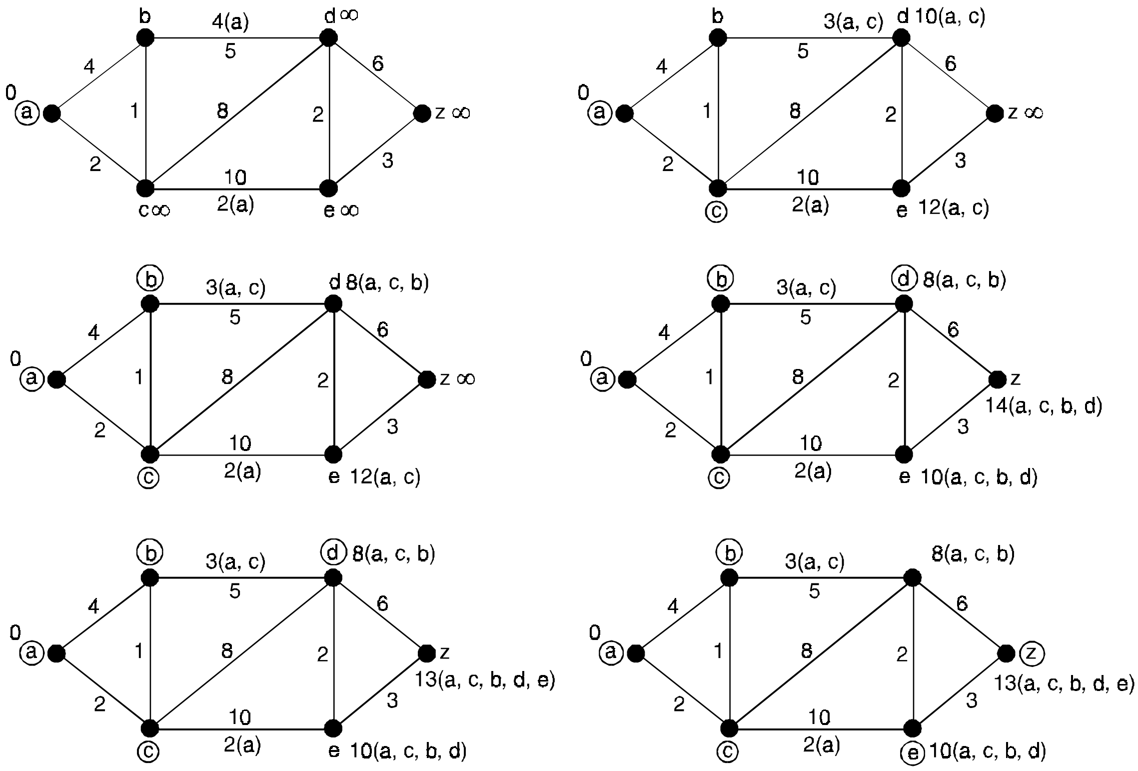


Fig. 4.6. Using Dijkstra's Algorithms to find a shortest path from *a* to *z*.

Theorem 4.1. *Dijkstra's algorithm finds the length of a shortest path between two vertices in a connected simple undirected weighted graph.*

Theorem 4.2. *Dijkstra's algorithm uses $O(n^2)$ operations (additions and comparisons) to find the length of a shortest path between two vertices in a connected simple undirected weighted graph with n vertices.*

Problem 4.5. *Apply Dijkstra's algorithm to the weighted graph G = (V, E) shown in Fig, (4.7) in order to find the shortest distance from vertex C (= v_0) to each of the other five vertices in G.*

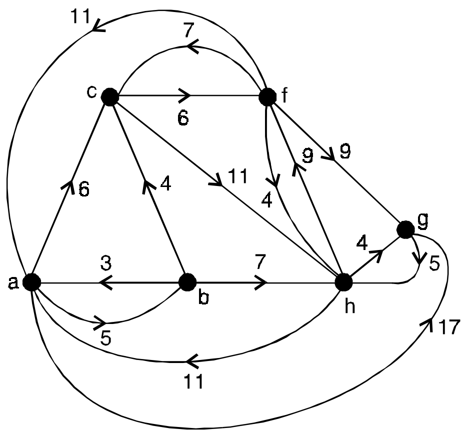


Fig. 4.7.

Solution. Initialization: ($i = 0$) Set $S_0 = \{c\}$. Label c with (0, –) and all other vertices in G with (∞, –).

First iteration : $(\bar{S}_0 = \{a, b, f, g, h\})$.

Here $i = 0$ in step (2) and we find, for example,

that $L(a) = \min \{L(a), L(c) + wt(c, a)\}$

$= \min \{\infty, 0 + \infty\} = \infty$

whereas $L(f) = \min \{L(f), L(c) + wt(c, f)\}$

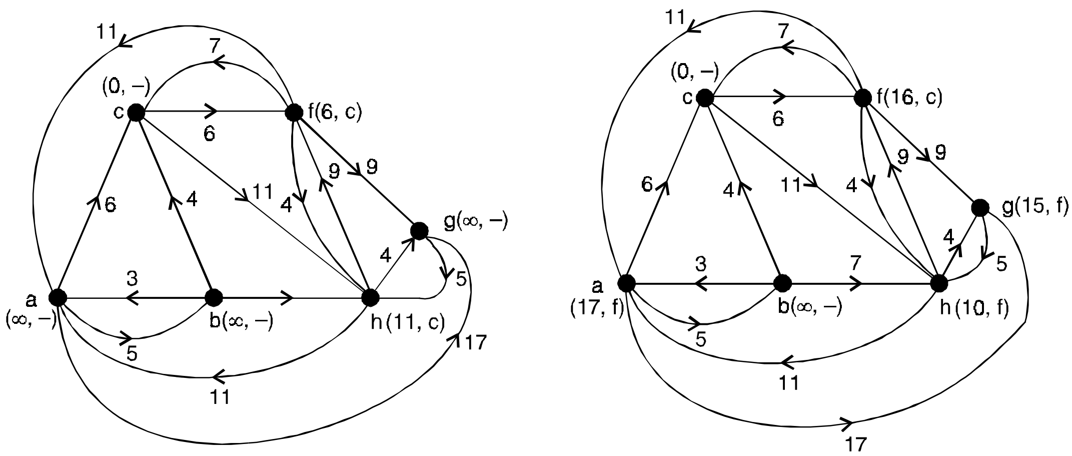
$= \min \{\infty, 0 + 6\} = 6.$

Similar calculations yield $L(b) = L(g) = \infty$ and $L(h) = 11$. So we label the vertex f with (6, c) and the vertex h with (11, c).

The other vertices in $\bar{S}_0$ remain labeled by (∞, –). See Fig. (4.8) (a).

In step (3) we see that f is the vertex v_1 in $\bar{S}_0$ closest to v_0, so we assign to S_1 the set $S_0 \cup \{f\} = \{c, f\}$ and increases the counter i to 1.

Since $i = 1 < 5$ (= 6 – 1), we return to step (2).



(a) (b)

Fig. 4.8.

Second iteration : $(\bar{S}_1 = \{a, b, g, h\})$.

Now $i = 1$ in step (2), and for each $v \in \bar{S}_1$ we set

$$L(v) = \min_{u \in S_1} \{L(v), L(u) + wt(u, v)\}$$

This yields

$L(a) = \min \{L(a), L(c) + wt(c, a), L(f) + wt(f, a)\}$

$= \min \{\infty, 0 + \infty, 6 + 11\} = 17.$

So vertex a is labeled {17, f).

In a similar manner, we find

$L(b) = \min \{\infty, 0 + \infty, 6 + \infty\} = \infty,$

$L(g) = \min\{\infty, 0 + \infty, 6 + 9\} = 15,$
$L(h) = \min\{11, 0 + 11, 6 + 4\} = 10.$

These results provide the labeling in Fig. (4.8) (*b*).

In step (3) we find that the vertex v_2 is h because $h \in \bar{S}_1$ and $L(h)$ is a minimum.

Then S_2 is assigned $S_1 \cup \{h\} = \{c, f, h\}$, the counter increased to 2, and since $2 < 5$, the algorithm directs us back to step (2).

Third iteration : $(\bar{S}_2 = \{a, b, g\})$. With $i = 2$ in step (2) the following are now computed.

$$L(a) = \min_{u \in S_2} \{L(a), L(u) + wt(u, a)\}$$
$$= \min\{17, 0 + \infty, 6 + 11, 10 + 11\} = 17$$

So the label on a is not changed,

$L(b) = \min\{\infty, 0 + \infty, 6 + \infty, 10 + \infty\} = \infty$

So the label on b remains ∞, and

$L(g) = \min\{15, 0 + \infty, 6 + 9, 10 + 4\} = 14 < 15.$

So the label on g is changed to $(14, h)$ because $14 = L(h) + wt(h, g)$.

Among the vertices in $\bar{S}_2$, g is the closest to v_0

Since $L(g)$ is a minimum.

In step (3), vertex v_3 is defined as g and $S_3 = S_2 \cup \{g\} = \{c, f, h, g\}$.

Then the counter i is increased to $3 < 5$, and we return to step (2).

Fourth iteration : $(\bar{S}_3 = \{a, b\})$. With $i = 3$, the following are determined in step (2),

$L(a) = 17$, $L(b) = \infty$.

Thus no labels are changed during this iteration. We set $v_4 = a$ and $S_4 = S_3 \cup \{a\} = \{c, f, h, g, a\}$ in step (3).

Then the counter i is increased to 4 (< 5), and we return to step (2).

Fifth iteration : $(\bar{S}_4 = \{b\})$. Here $i = 4$ in step (2), we find $L(b) = L(a) + wt(a, b) = 17 + 5 = 22$.

Now the label on b is changed to $(22, a)$.

Then $v_5 = b$ in step (3), S_5 is set to $\{c, f, h, g, a, b\}$ and i is incremented to 5.

But now that $i = 5 = |v| - 1$, the process terminates. We reach the labeled graph shown in Fig. (4.9).

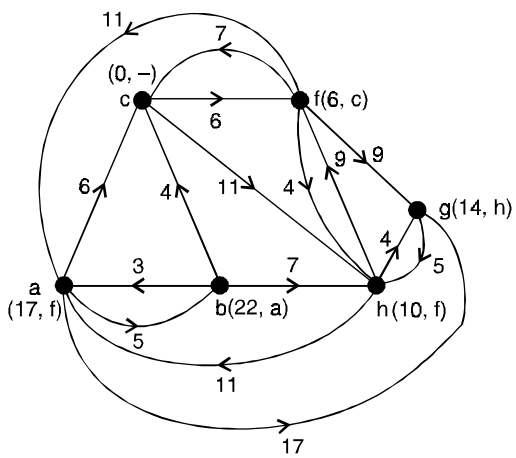


Fig. 4.9.

From the labels in Fig. (4.9) we have the following shortest distances from c to the other five vertices in G:

(*i*) $d(c, f) = L(f) = 6$ (*ii*) $d(c, h) = L(h) = 10$

(*iii*) $d(c, g) = L(g) = 14$ (*iv*) $d(c, a) = L(a) = 17$

(*v*) $d(c, b) = L(b) = 22$

To determine, for example, a shortest directed path from c to b, we start at vertex b, which is labeled $(22, a)$.

Hence a is the predecessor of b on this shortest path.

The label on a is $(17, f)$, so f precedes a on the path.

Finally, the label on f is $(6, c)$, so we are back at vertex c, and the shortest directed path from c to b determined by the algorithm is given by the edges (c, f), (f, a), and (a, b).

Problem 4.6. *Obtain the shortest distance and the shortest path from vertex 1 to vertex 7 in the network shown in Fig. (4.10).*

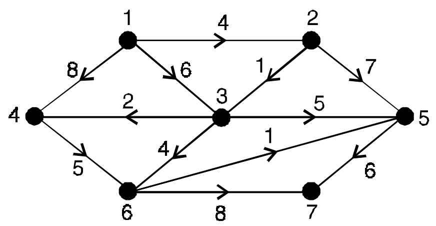


Fig. 4.10.

Solution. **First iteration :** For this iteration, $P = \{1\}$ with $P_1 = 0$ and $t_j = q_{1j}$ for $j = 2, 3, 4, 5$.

By examining the figure, we find that, the edges from 1 to 2 has weight 4, therefore $t_2 = 4$ and so on.

Since there is no edge from 1 to 5, we set $t_5 = \infty$ and so on.

$t_2 = 4$, $t_3 = 6$, $t_4 = 8$, $t_5 = \infty$, $t_6 = \infty$, $t_7 = \infty$.

Step 1 : We note that t_j is minimum for $j = 2$.

Therefore, we adjoin 2 to P, so that $P = \{1, 2\}$.

Also we label the arc (1, 2).

Step 2 : We have $P = \{1, 2\}$ and $t_2 = 4$ (from step 1)

We take new $t_3 = \min \{t_3, t_2 + q_{23}\}$

$= \min \{6, 4 + 1\} = 5,$

new $t_4 = \min \{t_4, t_2 + q_{24}\}$

$= \min \{8, 4 + \infty\} = 8,$

new $t_5 = \min \{t_5, t_2 + q_{25}\}$

$= \min \{\infty, 4 + 7\} = 11,$

new $t_6 = \min \{t_6, t_2 + q_{26}\} = \infty$

new $t_7 = \min \{t_7, t_2 + q_{27}\} = \infty.$

Second iteration

For this iteration, $P = \{1, 2\}$, and $P_1 = 0$, $P_2 = t_2 = 4$.

Also, $t_3 = 5$, $t_4 = 8$, $t_5 = 11$, $t_6 = \infty$, $t_7 = \infty$

Step 1 : We note that t_j, $j > 2$, is minimum for $j = 3$. Therefore, we adjoin 3 to P, so that P = {1, 2, 3}. Also, we label the arc (2, 3).

Step 2 : We have P = {1, 2, 3}, $t_2 = 4$, $t_3 = 5$,

We take new $t_4 = \min\{t_4, t_3 + q_{34}\}$

$= \min\{8, 5 + 2\} = 7,$

new $t_5 = \min\{t_5, t_3 + q_{35}\}$

$= \min\{11, 5 + 5\} = 10,$

new $t_6 = \min\{t_6, t_3 + q_{36}\}$

$= \min\{\infty, 5 + 4\} = 9,$

new $t_7 = \min\{\infty, 5 + q_{37}\} = \infty.$

Third iteration

For this iteration, P = {1, 2, 3}, $P_1 = 0$, $P_2 = 4$ and $P_3 = t_3 = 5$. Also, $t_4 = 7$, $t_5 = 10$, $t_6 = 9$, $t_7 = \infty$.

Step 1 : We note that t_j, $j > 3$, is minimum for $j = 4$. We adjoin 4 to P, so that P = {1, 2, 3, 4}. Also, we label the arc (3, 4).

Step 2 : We have P = {1, 2, 3, 4}, $t_2 = 4$, $t_3 = 5$, $t_4 = 7$.

We take new $t_5 = \min\{t_5, t_4 + q_{45}\}$

$= \min\{10, 7 + \infty\} = 10$

new $t_6 = \min\{t_6, t_4 + q_{46}\}$

$= \min\{9, 7 + 5\} = 9$

new $t_7 = \min\{t_7, t_4 + q_{47}\} = \infty$

Fourth iteration

For this iteration, P = {1, 2, 3, 4}, $P_1 = 0$, $P_2 = 4$, $l_3 = 5$ and $P_4 = t_4 = 7$. Also $t_5 = 10$, $t_6 = 9$, $t_7 = \infty$.

Step 1 : We note that t_j, $j > 4$, is minimum for $j = 6$. We adjoin 6 to P, so that P = {1, 2, 3, 4, 6}, $t_2 = 4$, $t_3 = 5$, $t_4 = 7$, $t_6 = 9$, $t_7 = \infty$.

We take new $t_5 = \min\{t_5, t_6 + q_{65}\}$

$= \min\{10, 9 + 1\} = 10,$

new $t_7 = \min\{t_7, t_6 + q_{67}\}$

$= \min\{\infty, 9 + 8\} = 17.$

Fifth iteration

For this iteration, P = {1, 2, 3, 4, 6}, $P_1 = 0$, $P_2 = 4$, $P_3 = 5$, $P_4 = 7$ and $P_6 = t_6 = 9$. Also, $t_5 = 10$, $t_7 = 17$.

Step 1 : We note that t_j is minimum for $j = 5$. We adjoin 5 to P, so that P = {1, 2, 3, 4, 6, 5}. Also, we label the arc (6, 5).

Step 2 : We have P = {1, 2, 3, 4, 6, 5}, $t_2 = 4$, $t_3 = 5$, $t_4 = 7$, $t_6 = 9$, $t_5 = 10$, $t_7 = 17$.

We take new $t_7 = \min\{t_7, t_5 + q_{57}\}$

$= \min\{17, 10 + 6\} = 16.$

Sixth iteration

For this iteration, P = {1, 2, 3, 4, 6, 5}, $P_1 = 0$, $P_2 = 4$, $P_3 = 5$, $P_4 = 7$, $P_P = 9$, $P_5 = t_5 = 10$. Also $t_7 = 16$.

Step 1 : Since there is now only one t_j left, namely t_7. We adjoin 7 to P, so that P = {1, 2, 3, 4, 6, 5, 7}. Also we label the arc (5, 7).

Step 2 : We have P = {1, 2, 3, 4, 6, 5, 7} which is the vertex set. We stop the process. We take $P_7 = t_7 = 16$.

Also, the shortest distance from vertex 1 to vertex 7 is 4 + 1 + 2 + 5 + 1 + 6 = 19.

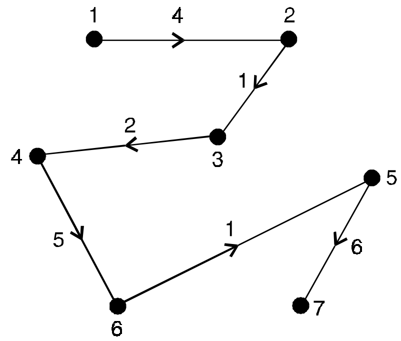


Fig. 4.11.

4.3. MINIMAL SPANNING TREES

4.3.1. Weighted graph

A weighted graph is a graph G in which each edge e has been assigned a non-negative number $w(e)$, called the weight (or length) of e. Figure (4.12) shows a weighted graph. The weight (or length) of a path in such a weighted graph G is defined to be the sum of the weights of the edges in the path. Many optimisation problems amount to finding, in a suitable weighted graph, a certain type of subgraph with minimum (or maximum) weight.

4.3.2. Minimal spanning tree

Let G be weighted graph. A minimal spanning tree of G is a spanning tree of G with minimum weight. The weighted graph G of Figure (4.12) shows six cities and the costs of laying railway links between certain pairs of cities. We want to set up railway links between the cities at minimum costs. The solution can be represented by a subgraph. This subgraph must be spanning tree since it covers all the vertices (so that each city is in the road system), it must be connected (so that any city can be reached from any other), it must have unique simple path between each pair of vertices.

Thus what is needed is a spanning tree the sum of whose weights is minimum, *i.e.*, a minimal spanning tree.

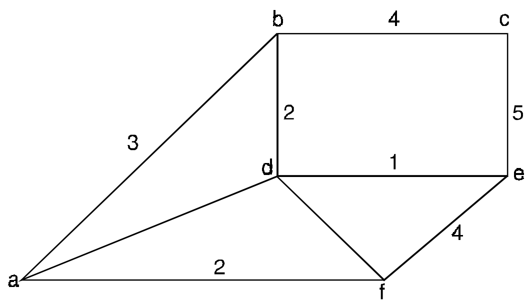


Fig. 4.12.

4.3.3. Algorithm for minimal spanning tree

There are several methods available for actually finding a minimal spanning tree in a given graph. Two algorithms due to Kruskal and Prim of finding a minimal spanning tree for a connected weighted graph where no weight is negative are presented below. These algorithms are example of greedy algorithms. A greedy algorithm is a procedure that makes an optimal choice at each of its steps without regard to previous choices.

4.3.4. Kruskal's algorithm

Kruskal's algorithm for finding a minimal spanning tree :

Input : A connected graph G with non-negative values assigned to each edge.

Output : A minimal spanning tree for G

Let G = (V, E) be graph and S = (V_s, E_s) be the spanning tree to be found from G. Let | V | = n and E = {e_1, e_2, ... e_m}. The stepwise algorithm is given below :

Method :

Step 1 : Select any edge of minimal value that is not a loop. This is the first edge of T. If there is more than one edge of minimal value, arbitrarily choose one of these edges.

i.e., select an edge e_1 from E such that e_1 has least weight. Replace E = E – {e_1} and E_s = {e_1}

Step 2 : Select any remaining edge of G having minimal value that does not form a circuit with the edges already included in T.

i.e., select an edge e_i from E such that e_i has least weight and that it does not form a cycle with members of E_s. Set E = E – {e_i} and $E_s = E_s \cup \{e_i\}$.

Step 3 : Continue step 2 until T contains n – 1 edges, where n is the number of vertices of G.

i.e., Repeat step 2 until | E_s | = | V | – 1.

Suppose that a problem calls for finding an optimal solution (either maximum or minimum).

Suppose, further, than an algorithm is designed to make the optimal choice from the available data at each stage of the process. Any algorithm based on such an approach is called a greedy algorithm.

A greedy algorithm is usually the first heuristic algorithm one may try to implement and it does lead to optimal solutions sometimes, but not always. Kruskal's algorithm is an example of a greedy algorithm that does, in fact, lead to an optimal solution.

Theorem 4.3. *Let G = (V, E) be a loop-free weighted connected undirected graph. Any spanning tree for G that is obtained by Kruskal's algorithm is optimal.*

Proof. Let | V | = n, and let T be a spanning tree for G obtained by Kruskal's algorithm.

The edges in T are labeled e_1, e_2, ... e_{n-1}, according to the order in which they are generated by the algorithm.

For each optimal tree T′ of G, define d(T′) = k if k is the smallest positive integer such that T and T′ both contain e_1, e_2, .., e_{k-1}, but $e_k \notin$ T′.

Let T_1 be an optimal tree for which $d(T_1) = r$ is maximal.

If $r = n$, then T = T_1 and the result follows.

Otherwise, $r \le n - 1$ and adding edge e_r (of T) to T_1 produces the cycle C, where there exists an edge e_r' if C that is in T_1 but not in T.

Start with tree T_1. Adding e_r to T_1 and deleting e_r', we obtain a connected graph with n vertices and $n - 1$ edges.

This graph is a tree, T_2. The weights of T_1 and T_2 satisfy $wt(T_2) = wt(T_1) + wt(e_r) - wt(e_r')$.

Following the selection of $e_1, e_2, ..., e_{r-1}$ in Kruskal's algorithm, the edge e_r is chosen so that $wt(e_r)$ is minimal and no cycle results when e_r is added to the subgraph H of G determined by $e_1, e_2, ..., e_{r-1}$.

Since e_r' produces no cycle when added to the subgraph H, by the minimality of $wt(e_r)$ it follows that $wt(e_r') \geq wt(e_r)$.

Hence $wt(e_r) - wt(e_r') \leq 0$, so $wt(T_2) \leq wt(T_1)$. But with T_1 optimal, we must have $wt(T_2) = wt(T_1)$, so T_2 is optimal.

The tree T_2 is optimal and has the edges $e_1, e_2, ..., e_{r-1}, e_r$ in common with T, so $d(T_2) \geq r + 1 > r = d(T_1)$, contradicting the choice of T_1.

Consequently, $T_1 = T$ and the tree T produced by Kruskal's algorithm is optimal.

Theorem 4.4. *Let G be a connected graph where the edges of G are labelled by non-negative numbers. Let T be an economy tree of G obtained from Kruskal's Algorithm. Then T is a minimal spanning tree.*

Proof. As before, for each edge e of G, let $C(e)$ denote the value assigned to the edge by the labelling.

If G has n vertices, an economy tree T must have $n - 1$ edges.

Let the edges $e_1, e_2, ..., e_{n-1}$ be chosen as in Kruskal's Algorithm. Then $C(T) = \sum_{i=1}^{n-1} C(e_i)$.

Let T_0 be a minimal spanning tree of G.

We show that $C(T_0) = C(T)$, and thus conclude that T is also minimal spanning tree.

If T and T_0 are not the same let e_i be the first edge of T not in T_0.

Add the edge e_i to T_0 to obtain the graph G_0.

Suppose $e_i = \{a, b\}$, then a path P from a to b exists in T_0 and so P together with e_i produces a circuit C in G_0.

Since T contains no citcuits, there must be an edge e_0 in C that is not in T.

The graph $T_1 = G_0 - e_0$ is also a spanning tree of G since T_1 has $n - 1$ edges.

Moreover, $C(T_1) = C(T_0) + C(e_i) - C(e_0)$.

However, we know that $C(T_0) \leq C(T_1)$ since T_0 was a minimal spanning tree of G.

Thus, $C(T_1) - C(T_0) = C(e_i) - C(e_0) \geq 0$.

Implies that $C(e_i) \geq C(e_0)$.

However, since T was constructed by Kruskal's algorithm e_i is an edge of smallest value that can be added to the edges $e_1, e_2, ..., e_{i-1}$ without producing a circuit. Also, if e_0 is added to the edges $e_1, e_2, ..., e_{i-1}$, no circuit is produced because the graph thus formed is a subgraph of the tree T_0.

Therefore, $C(e_i) = C(e_0)$, so that $C(T_1) = C(T_0)$.

We have constructed from T_0 a new minimal spanning tree T_1 such that the number of edges common to T_1 and T exceeds the number of edges common to T_0 and T by one edge, namely e_i.

Repeat this procedure, to construct another minimal spanning tree T_2 with one more edge in common with T than was in common between T_1 and T.

By continuing this procedure, we finally arrive at a minimal spanning tree with all edges in common with T, and thus we conclude that T is itself a minimal spanning tree.

Problem 4.7. *Using Kruskal's algorithm, find the minimum spanning tree for the weighted graph of the Fig. (4.13).*

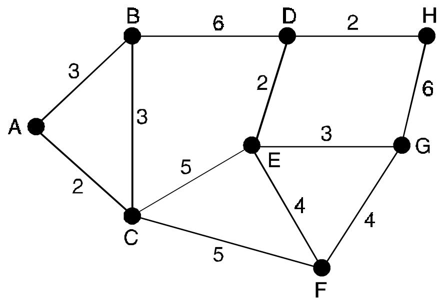


Fig. 4.13.

Solution. Let S = (V_s, E_s) be the spanning tree to be found from G.

Initialize, there are eight nodes so the spanning tree will have seven arcs.

The iterations of algorithm applied on the graph are given below and it runs at the most seven times.

The number indicates iteration number.

1. Since arcs AC, ED, and DH have minimum weight 2. Since they do not form a cycle, we select all of them and E_s = {(A, C), (E, D), (D, H)} and E = E – ({A, C), (E, D), (D, H)}.

2. Next arcs with minimum weights 3 are AB, BC, and EG. We can select only one of the AB and BC. Also we can select EG.

Therefore, E_s = ({A, C), (E, D), (D, H), (A, B), (E, G)} and E = E – {(A, B), (E, G)}

3. Next arcs with minimum weights 4 are EF and FG. We can select only one of them.

Therefore, E_s = {(A, C), (E, D), (D, H), (A, B), (E, G), (F, G)} and E = E – {(F, G)}.

4. Next arcs with minimum weights 5 are CE and CF. We can select only one of them.

Therefore, E_s = ({A, C), (E, D), (D, H), (A, B), (E, G), (F, G), (C, E)} and E = E – {(C, E)}.

Since number of edges in E_s is seven process terminates here. The spanning tree so obtained is shown in the Fig. (4.14).

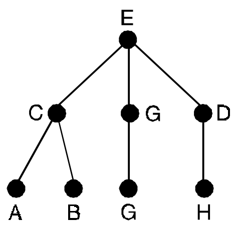


Fig. 4.14.

Problem 4.8. *Show how Kruskal's algorithm find a minimal spanning tree for the graph of Fig. (4.15).*

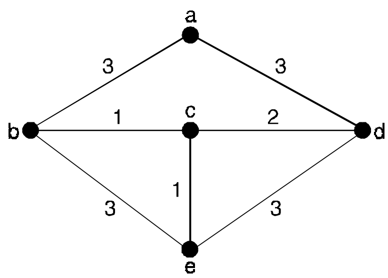


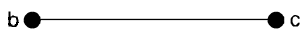
Fig. 4.15.

Solution. We collect the edges with their weights into a table

Edge	*Weight*
(b, c)	1
(c, e)	1
(c, d)	2
(a, b)	3
(e, d)	3
(a, d)	4
(b, e)	4

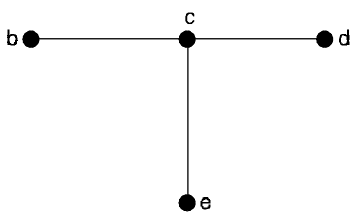
The steps of finding a minimal spanning tree are shown below.

1. Choose the edge (b, c) as it has a minimal weight

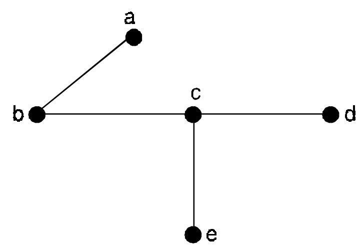


2. Add the next edge (c, e)

3. Add the edge (c, d)



4. Add the edge (b, a)



Since vertices are 5 and we have choosen 4 edges, we stop the algorithm and the minimal spanning tree is produced.

Problem 4.9. *Show how Kruskal's algorithm find a minimal spanning tree of the graph of Fig. (4.16).*

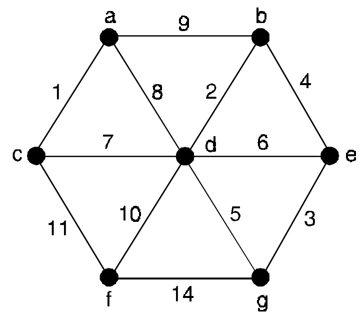


Fig. 4.16.

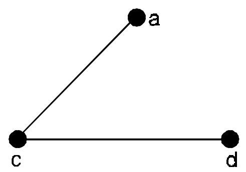
Solution. We collect the edges with their weights into a table.

Edge	*Weight*
(a, c)	1
(b, d)	2
(e, g)	3
(b, e)	4
(d, g)	5
(d, e)	6
(d, c)	7
(a, d)	8
(a, b)	9
(d, f)	10
(c, f)	11
(f, g)	14

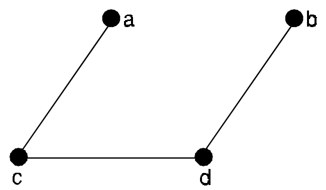
The steps of finding a minimal spanning tree are shown below :

1. Choose the edge (a, c) as it has minimal weight

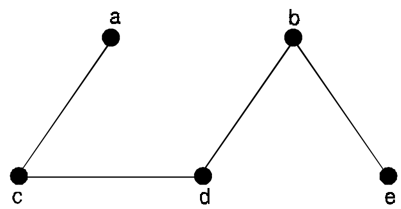
2. Add the next edge (c, d)



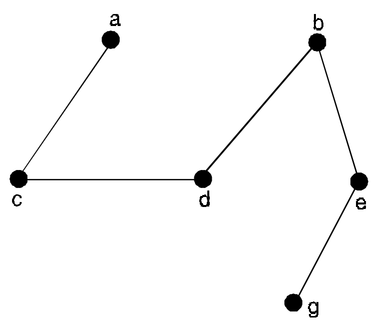
3. Add the edge (d, b)



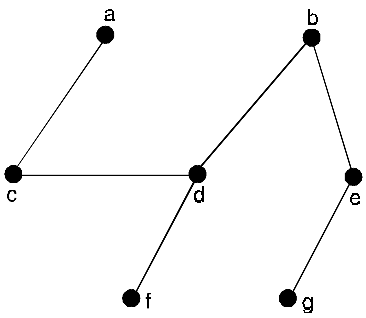
4. Add the edge (b, e)



5. Add the edge (e, g)



6. Add the edge (d, f)



Since vertices are 7 and we have chosen 6 edges, we stop the algorithm and the minimal spanning tree is produced.

Problem 4.10. *Use Kruskal's algorithm to find a minimum spanning tree in the weighted graph shown in Fig. (4.17).*

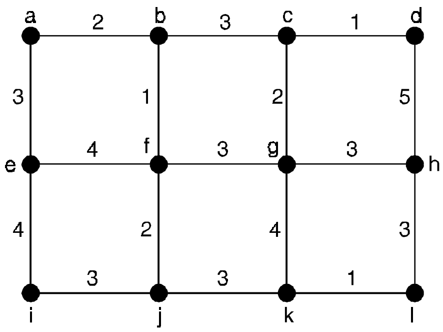


Fig. 4.17.

Solution. A minimum spanning tree and the choices of edges at each stage of Kruskal's algorithm are shown in Fig. (4.18)

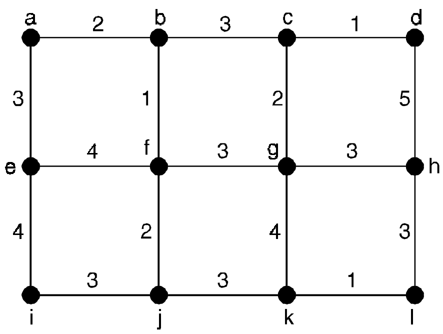


Fig. 4.18.

Choice	*Edge*	*Weight*
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
		Total : 24

Problem 4.11. *Determine a railway network of minimal cost for the cities in Fig. (4.19).*

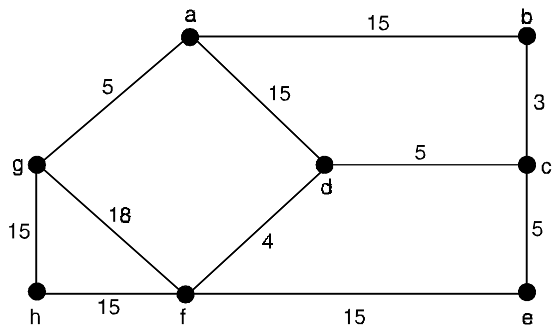


Fig. 4.19.

Solution. We collect lengths of edges into a table :

Edge	*Cost*
$\{b, c\}$	3
$\{d, f\}$	4
$\{a, g\}$	5
$\{c, d\}$	5
$\{c, e\}$	5
$\{a, b\}$	15
$\{a, d\}$	15
$\{f, h\}$	15
$\{g, h\}$	15
$\{e, f\}$	15
$\{f, g\}$	18

1. Choose the edges $\{b, c\}$, $\{d, f\}$, $\{a, g\}$, $\{c, d\}$, $\{c, e\}$
2. Then we have options : we may choose only one of $\{a, b\}$ and $\{a, d\}$ for the selection of both creates a circuit. Suppose that we choose $\{a, b\}$.
3. Likewise we may choose only one of $\{g, h\}$ and $\{f, h\}$. Suppose we choose $\{f, h\}$.
4. We then have a spanning tree as illustrated in Fig. (4.20).

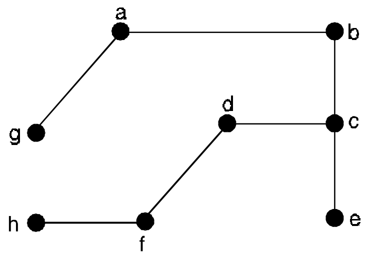


Fig. 4.20.

The minimal cost for construction of this tree is

$$3 + 4 + 5 + 5 + 5 + 15 + 15 = 52.$$

Problem 4.12. *Apply Kruskal's algorithm to the graph shown in Fig. (4.21).*

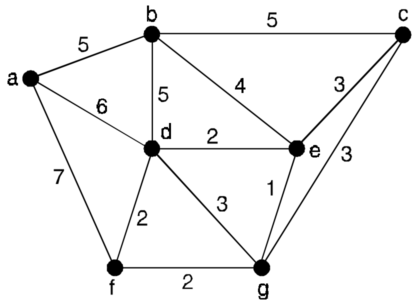


Fig. 4.21.

Solution. Initialization : ($i = 1$) since there is a unique edge-namely, $\{e, g\}$, of smallest weight 1, start with T = $\{\{e, g\}\}$. (T starts as a tree with one edge, and after each iteration it grows into a larger tree or forest. After the last iteration the subgraph T is an optimal spanning tree for the given graph G).

First iteration

Among the remaining edges in G, three have the next smallest weight 2. Select $\{d, f\}$, which satisfies the conditions in step (2).

Now T is the forest $\{\{e, g\}, \{d, f\}\}$, and i is increased to 2. With $i = 2 < 6$, return to step (2).

Second iteration

Two remaining edges have weight 2. Select $\{d, e\}$.

Now T is the tree $\{\{e, g\}, \{d, f\}, \{d, e\}\}$, and i increases to 3. But because $3 < 6$, the algorithm directs us back to step (2).

Third iteration

Among the edges of G that are not in T, edge {*f*, *g*} has minimal weight 2.

However, if this edge is added to T, the result contains a cycle, which destroys the tree structure being sought.

Consequently, the edges {*c*, *e*}, {*c*, *g*} and {*d*, *g*} are considered.

Edge {*d*, *g*} brings about a cycle, but either {*c*, *e*} or {*c*, *g*} satisfies the conditions in step (2).

Select {*c*, *e*}. T grows to {{*e*, *g*}, {*d*, *f*}, {*d*, *e*}, {*c*, *e*}} and *i* is increased to 4.

Returning to step (2), we find that the fourth and fifth iterations provide the following.

Fourth iteration

T = {{*e*, *g*}, {*d*, *f*}, {*d*, *e*}, {*c*, *e*}, {*b*, *e*}}, *i* increases to 5.

Fifth iteration

T = {{*e*, *g*}, {*d*, *f*}, {*d*, *e*}, {*c*, *e*}, {*b*, *e*}, {*a*, *b*}}.

The counter *i* now becomes 6 = (number of vertices in G) – 1.

So T is an optimal tree for graph G and has weight

$$1 + 2 + 2 + 3 + 4 + 5 = 17$$

Fig. (4.22) shows this spanning tree of minimal weight.

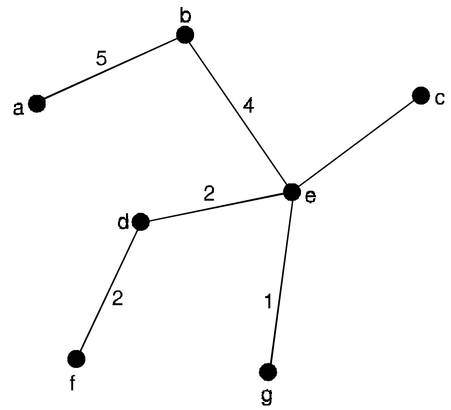


Fig. 4.22.

Problem 4.13. *Using the Kruskal's algorithm, find a minimal spanning tree of the weighted graph shown below :*

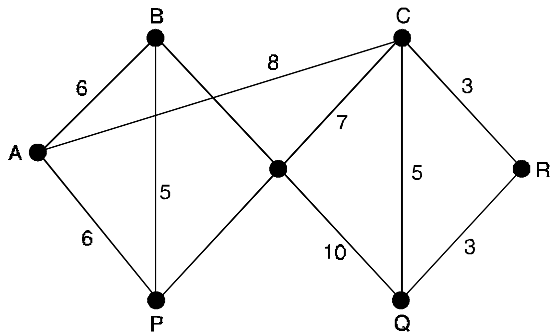


Fig. 4.23.

Solution. We observe that the given graph has 6 vertices, hence a spanning tree will have 5 edges. Let us put the edges of the graph in the non-decreasing order of their weights and successively select 5 edges in such a way that no circuit is created.

Edges	CR	QR	BP	CQ	AB	AP	CP	AC	BQ
Weight	3	3	5	5	6	6	7	8	10
Select	Yes	Yes	Yes	No	Yes	No	Yes		

Observe that CQ is not selected because CR and QR have already been selected and the selection of CQ would have created a circuit. Further, AP is not selected because it would have created a circuit along with BP and AB which have already been selected we have stopped the process when exactly 5 edges are selected.

Thus, a minimal spanning tree of the given graph contains the edges CR, QR, BP, AB, CP. This tree as shown in Fig. (4.24). The weight of this tree is 24 units.

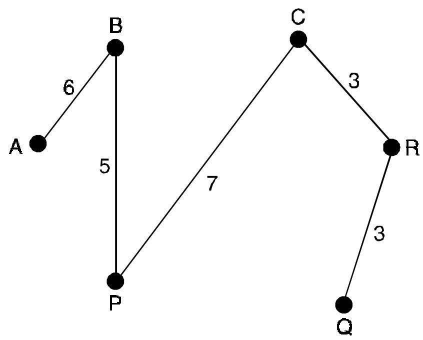


Fig. 4.24.

Problem 4.14. *Eight cities A, B, C, D, E, F, G, H are required to be connected by a new railway network. The possible tracks and the cost of involved to lay them (in crores of rupees) are summarized in the following table :*

Track between	*Cost*	*Track between*	*Cost*
A and B	*155*	*D and F*	*100*
A and D	*145*	*E and F*	*150*
A and G	*120*		
B and C	*145*	*F and G*	*140*
C and D	*150*	*F and H*	*150*
C and E	*95*	*G and H*	*160*

Determine a railway network of minimal cost that connects all these cities.

Solution. Let us first prepare a graph whose the vertices represent the cities, edges represent the possible tracks and weights represent the cost. The graph is as shown in Fig. (4.25).

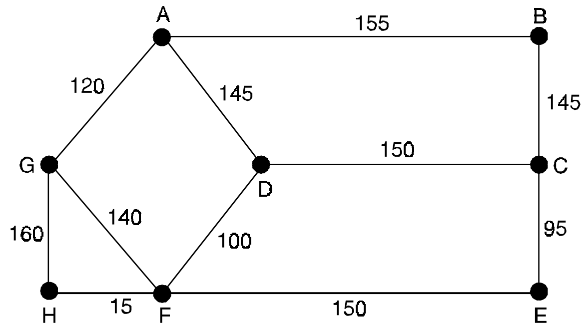


Fig. 4.25.

A minimal spanning tree of this graph represents the required network. Since there are eight vertices, seven edges should be there in a minimal spanning tree.

Let us put the edges of the graph in the non-decreasing order of their weights and select seven edges one by one in such a way that no circuit is created.

Edges	CE	DF	AG	FG	AD	BC	CD	EF	FH	AB	GH
Weight	95	100	120	140	145	145	150	150	150	155	160
Select	Yes	Yes	Yes	Yes	No	Yes	Yes	No	Yes.		

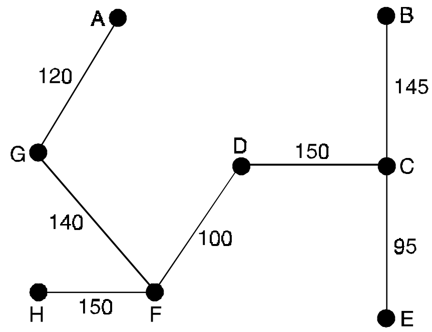


Fig. 4.26.

Thus, a minimal spanning tree of the given graph consists of the branches CE, DF, AG, FG, BC, CD, FH.

This tree represents the required railway network. The network is shown in Fig. (4.26). The cost involved is $95 + 100 + 120 + 140 + 145 + 150 + 150 = 900$ (in crores of rupees).

Problem 4.15. *Using the Kruskal's algorithm, find a minimal spanning tree for the weighted graph shown below :*

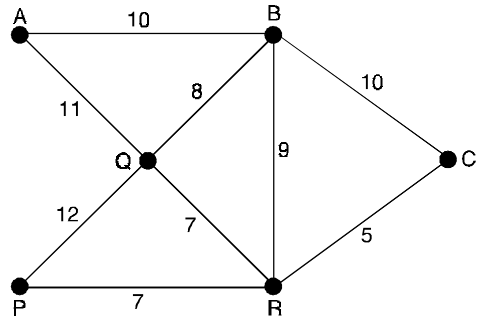


Fig. 4.27.

Solution. The given graph has 6 vertices and therefore a spanning tree will have 5 edges.

Let us put the edges of the graph in the non-decreasing order of their weights and go on selecting 5 edges one by one in such a way that no circuit is created.

Edges	CR	PR	QR	BQ	BR	AB	BC	AR	PQ
Weight	5	7	7	8	9	10	10	11	12
Select	Yes	Yes	Yes	Yes	No	Yes			

Thus, a minimal spanning tree of the given graph contains the edges CR, PR, QR, BQ, AB. The tree is shown in Fig. (4.28). The weight of the tree is 37 units.

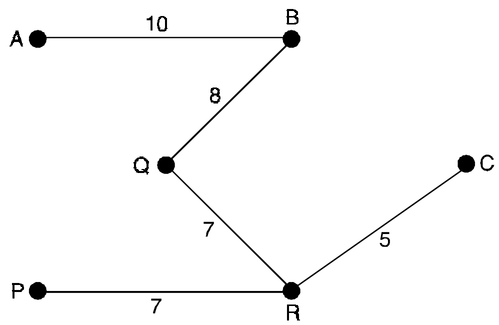


Fig. 4.28.

4.3.5. Prim's algorithm

Let G = (V, E) be a connected graph with | V | = n. To find the adjacency matrix for G. Now proceed according to the following steps :

Step 1 : Select a vertex $v_1 \in$ V and arrange the adjacency matrix of the graph in such a way that the first row and first column of the matrix corresponds to V_1.

Step 2 : Choose a vertex v_2 of V such that $(v_1, v_2) \in$ E. Merge v_1 and v_2 into a new vertex, call it $v_m^{\,i}$ and drop v_2. Replace v_1 by $v_m^{\,i}$ in the graph. Find the new adjacency matrix corresponding to this new quotient graph.

Step 3 : While merging select an edge from those edges which are going to be removed (or merged with other edge) from the graph. Keep a record of it.

Step 4 : Repeat steps 1, 2 and 3 until all vertices have been merged into one vertex.

Step 5 : Now construct a tree from the edges, collected at different iterations of the algorithm.

The same Prim's algorithm with little modification can be used to find a minimum spanning tree for a weighted graph. The stepwise algorithm is given below.

Let G = (V, E) be graph and S = (V_s, E_s) be the spanning tree to be found from G.

Step 1 : Select a vertex v_1 of V and initialize

$$V_s = \{v_1\} \text{ and } E_s = \{\}$$

Step 2 : Select a nearest neighbour of v_i from V that is adjacent to same $v_j \in V_s$ and that edge (v_i, v_j) does not form a cycle with members edge of E_s.

$$\text{Set} \quad V_s = V_s \cup \{v_i\} \text{ and}$$
$$E_s = E_s \cup \{(v_i, v_j)\}$$

Step 3 : Repeat step 2 until $|\, E_s \,| = |\, V \,| - 1$.

Theorem 4.5. *Prim's algorithms produces a minimum spanning tree of a connected weighted graph.*

Proof. Let G be a connected weighted graph.

Suppose that the successive edges chosen by Prim's algorithm are $e_1, e_2, ..., e_{n-1}$.

Let S be the tree with $e_1, e_2, ..., e_{n-1}$ as its edges, and let S_k be the tree with $e_1, e_2, ..., e_k$ as its edges.

Let T be a minimum spanning tree of G containing the edges $e_1, e_2, ..., e_k$, where k is the maximum integer with the property that a minimum spanning tree exists containing the first k edges chosen by Prim's algorithm. The theorem follows if we can show that S = T.

Suppose that S ≠ T, so that $k < n - 1$.

Consequently, T contains $e_1, e_2, ..., e_k$, but not e_{k+1}.

Consider the graph made up of T together with e_{k+1}.

Since this graph is connected and has n edges, too many edges to be a tree, it must contain a simple circuit.

This simple circuit must contain e_{k+1} since there was no simple circuit in T.

Furthermore, there must be an edge in the simple circuit that does not belong to S_{k+1} since S_{k+1} is a tree.

By starting at an end point of e_{k+1} that is also an endpoint of one of the edges $e_1, ..., e_k$, and following the circuit until it reaches an edge not in S_{k+1}, we can find an edge e not in S_{k+1} that has an end point that is also an end point of one of the edges $e_1, e_2, ..., e_k$.

By deleting e from T and adding e_{k+1}, we obtain a tree T′ with $n - 1$ edges (it is a tree since it has no simple circuits).

Note that the tree T′ contains $e_1, e_2, ..., e_k, e_{k+1}$.

Furthermore, since e_{k+1} was chosen by Prim's algorithms at the (k + 1)st step, and e was also available at that step, the weight of e_{k+1} is less than or equal to the weight of e.

From this observation it follows that T′ is also a minimum spanning tree, since the sum of the weights of its edges does not exceed the sum of the weights of the edges of T.

This contradicts the choice of k as the maximum integer so that a minimum spanning tree exists containing $e_1, ..., e_k$.

Hence, $k = n - 1$ and S = T.

It follows that Prim's algorithm produces a minimum spanning tree.

Problem 4.16. *Find the minimum spanning tree for the weighted graph of the Figure (4.29).*

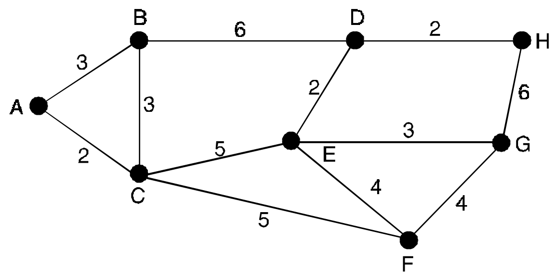


Fig. 4.29.

Solution. Let is begin with the node A of the graph.

Let S = (V_s, E_s) be the spanning tree to be found from G.

Initialize V_s = {A} and E_s = { }.

There are eight nodes so the spanning tree will have seven arcs.

The iterations of algorithm applied on the graph are given below. The number indicates iteration number.

1. Nodes B and C are neighbours of A. Since node C is nearest to the node A we select C.

   Thus, we have V_s = {A, C} and E_s = {(A, C)}.

2. Now node B is neighbour of both A and C and C has nodes E and F as its neighbour. We have AB = 3, CB = 3, CE = 5, and CF = 5.

   Thus, the nearest neighbour is B. We can select either AB or CB. We select CB.

   Therefore, V_s = {A, C, B} and E_s = {(A, C), (C, B)}.

3. Now D, E, F are neighbour of nodes in V_s. An arc AB is still to be considered. This arc forms cycle with arcs AC and CB already in E_s so it cannot be selected.

   Thus, we have to select from BD = 6, CE = 5, CF = 5.

   We may take either CE or CF. We select CF.

   Therefore, V_s = {A, C, B, F} and E_s = {(A, C), (C, B), (C, F)}

4. Now we have to select an arc from BD = 6, CE = 5, FE = 4, FG = 4. We select FE.

   Therefore, V_s = {A, C, B, F, E} and E_s = {(A, C), (C, B), (C, F), (F, E)}.

5. The selection of arc CE is ruled out as it forms a cycle with the edges CF and FE.
   Thus, we have to select an arc from BD = 6, ED = 2, FG = 4. We select ED.
   Therefore, V_s = {A, C, B, F, E, D} and
   E_s = {(A, C), (C, B), (C, F), (F, E), (E, D)}.
6. Now BD is ruled out as it forms cycle with CB, CF, FE and ED.
   Thus we have to consider DH = 2, EG = 3, FG = 4. We select DH.
   Therefore V_s = {A, C, B, F, E, D, H} and
   E_s = {(A, C), (C, B), (C, F), (F, E), (E, D), (D, H)}.
7. Now left over arcs are EG = 3, HG = 6, FG = 4. We select EG.
   Therefore, V_s = {A, C, B, F, E, D, H, G} and
   E_s = {(A, C), (C, B), (C, F), (F, E), (E, D), (D, H), (E, G)}.
   Since number of edges in E_s is seven process terminates here. The spanning tree so obtained is shown in the Fig. (4.30).

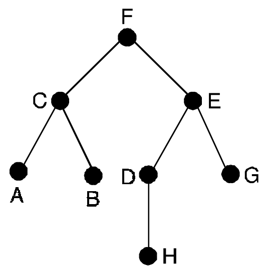


Fig. 4.30.

Problem 4.17. *Find a minimum spanning tree from the graph of the Fig. (4.31).*

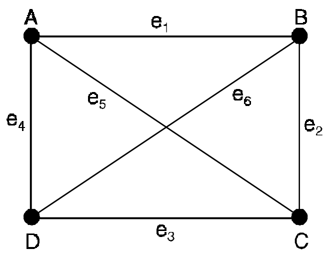


Fig. 4.31.

Solution. The working of the procedure is shown in the following table. The first row of the table contains the adjacency matrix of the given graph. If there are *n* nodes in a graph then merging process will continue up to *n*th iterations.

Matrix	*Nodes merged*	*Next node*	*Edge kept*
$\begin{array}{c c} & \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$		B	
$\begin{array}{c c} & \begin{array}{ccc} A_m^1 & C & D \end{array} \\ \begin{array}{c} A_m^1 \\ C \\ D \end{array} & \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \end{array}$	$A^1{}_m = \{A, B\}$	C	(A, B)
$\begin{array}{c c} & \begin{array}{cc} A_m^2 & D \end{array} \\ \begin{array}{c} A_m^2 \\ D \end{array} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{array}$	$A^2{}_m = \{A, B, C\}$	D	(A, C)
$\begin{array}{c c} & A_m^3 \\ A_m^3 & [0] \end{array}$	$A^3{}_m = \{A, B, C, D\}$		(A, D)

Therefore, we get three arcs in the process. The tree constructed with these arcs is shown in the Fig. (4.32)

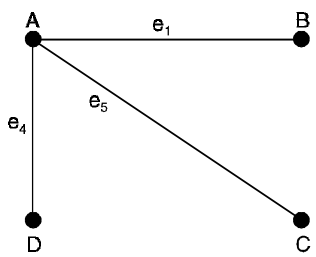


Fig. 4.32.

Problem 4.18. *Using Prim's algorithm, find a minimal spanning tree for the weighted graph shown in Fig. (4.33).*

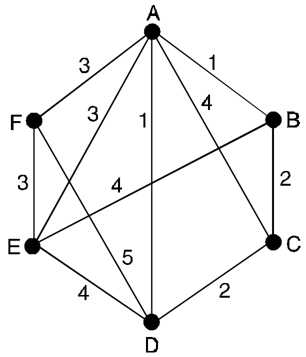


Fig. 4.33.

Solution. The given graph has 6 vertices.

Therefore, a minimal spanning tree thereof has 5 vertices.

	A	*B*	*C*	*D*	*E*	*F*
A	–	1	4	1	3	3
B	1	–	2	∞	4	∞
C	4	2	–	2	∞	∞
D	1	∞	2	–	4	5
E	3	4	∞	4	–	3
F	3	∞	∞	5	3	–

Now we examine the A-row and pick the smallest entry 1 which corresponds to two edges (A, B) and (A, D).

Let us choose the first of these, namely (A, B). By examining the A- and B-rows, we find that the vertex other than A and B which corresponds to the smallest entry is C.

Thus, C is closest to the edge (A, B). Let us connect C to (A, B).

Next, we examine the C-row and pick the smallest entry 2. The edge (C, D) is one of the corresponding edges. By examining the C- and D-rows, we find that the vertex other than C and D which corresponds to the smallest entry and which does not produce a circuit is E.

Thus, E is closest to the edge (C, D). Let us connect E to (C, D).

The construction at this stage shows that the edges (A, B), (B, C), (C, D), (D, E) belong to a minimal tree. The vertex left over is F, which is joined to A, D and E in the given graph.

Among the edges that contain F, the edges FA and FE have minimum weight.

Therefore, we can include either of these in the minimal spanning tree.

Thus, for the given graph we get two minimal spanning trees shown in Fig. (4.34)(*a*), (*b*)

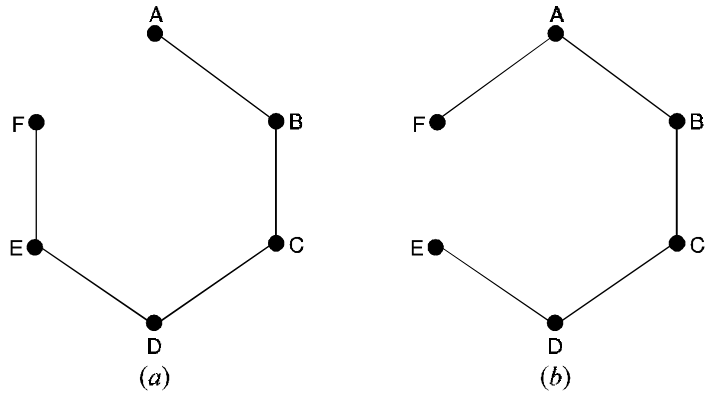


Fig. 4.34.

Problem 4.19. *Find the minimal spanning tree of the weighted graph of Fig. (4.35), using Prim's algorithm.*

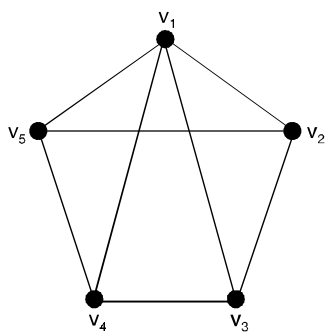
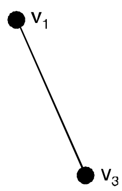
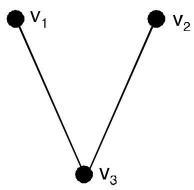


Fig. 4.35.

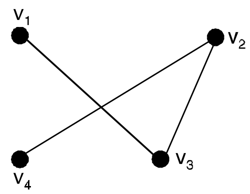
Solution. 1. We choose the vertex v_1. Now edge with smallest weight incident on v_1 is (v_1, v_3), so we choose the edge or (v_1, v_5).



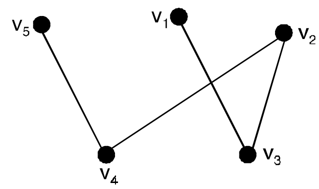
2. Now $w(v_1, v_2) = 4$, $w(v_1, v_4) = 3$, $w(v_1, v_5) = 4$, $w(v_1, v_2) = 2$ and $w(v_3, v_4) = 3$. We choose the edge (v_3, v_2) since it is minimum.



3. Again $w(v_1, v_5) = 3$, $w(v_2, v_4) = 1$ and $w(v_3, v_4) = 3$. We choose the edge (v_2, v_4).



4. Now we choose the edge (v_4, v_5). Now all the vertices are covered.
   The minimal spanning tree is produced.



Problem 4.20. *Use Prim's algorithm to design a minimum-cost communications network connecting all the computers represented by the graph in Fig. (4.36).*

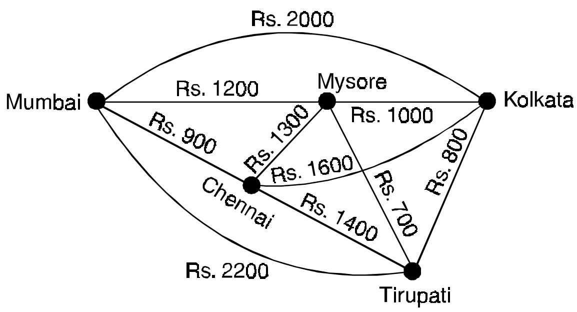


Fig. 4.36.

Solution. We solve this problem by finding a minimum spanning tree in the graph in Fig. (4.36).

Prim's algorithm is carried out by choosing an initial edge of minimum weight and successively adding edges of minimum weight that are incident to a vertex in the tree and that do not form simple circuits.

The edges in colors in Fig. (4.37) show a minimum spanning tree produced by Prim's algorithm, with the choice made at each step displayed.

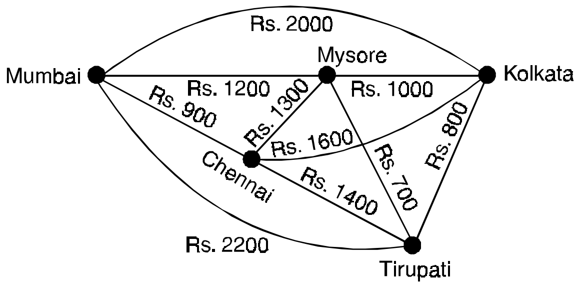


Fig. 4.37.

Choice	*Edge*	*Cost*
1	{Mysore, Tirupati}	Rs. 700
2	{Tirupati, Kolkata}	Rs. 800
3	{Mysore, Mumbai}	Rs. 1200
4	{Mumbai, Chennai}	Rs. 900
		Total Rs. 3600

Problem 4.21. *Use Prim's algorithm to find a minimum spanning tree in the graph shown in Fig. (4.38).*

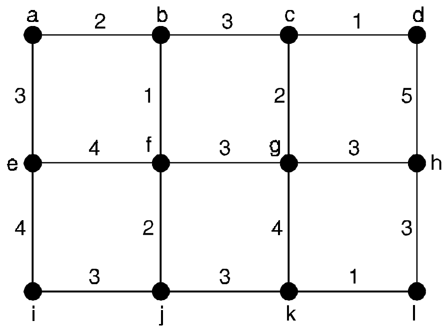


Fig. 4.38.

Solution. A minimum spanning tree constructed using Prim's algorithm is shown in Fig. (4.39). The successive edges chosen are displayed.

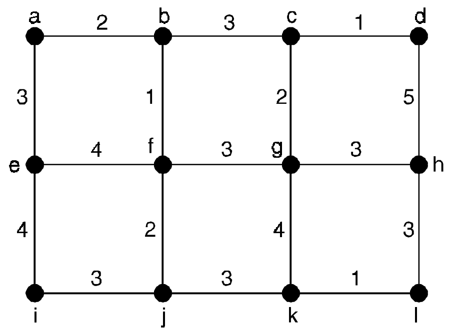


Fig. 4.39.

Choice	*Edge*	*Weight*
1	$\{b, f\}$	1
2	$\{a, b\}$	2
3	$\{f, j\}$	2
4	$\{a, e\}$	3
5	$\{i, j\}$	3
6	$\{f, g\}$	3
7	$\{c, g\}$	2
8	$\{c, d\}$	1
9	$\{g, h\}$	3
10	$\{h, l\}$	3
11	$\{k, l\}$	1
		Total : 24

Problem 4.22. *Using Prim's algorithm, find a minimal spanning tree for the weighted graph shown in Fig. (4.40).*

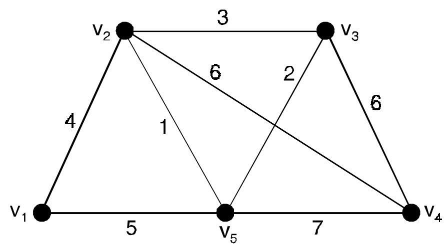


Fig. 4.40.

Solution. We observe that the graph has 5 vertices. Therefore, a minimal spanning tree will have 4 edges.

Let us tabulate the vertices of the edges between every pair of vertices as shown below :

	v_1	v_2	v_3	v_4	v_5
v_1	–	4	∞	∞	5
v_2	4	–	3	6	1
v_3	∞	3	–	6	2
v_4	∞	6	6	–	7
v_5	5	1	2	7	–

Now, let us start with the first row (v_1-row) and pick the smallest entry therein. This is 4 which corresponds to the edge (v_1, v_2).

By examining all the entries in v_1- and v_2-rows, we find that the vertex other than v_1 and v_2 which corresponds to the smallest entry is v_5.

Thus, v_5 is closest to the edge (v_1, v_2). Let us connect v_5 to the edge (v_1, v_2).

Let us now examine the v_5-row and note that the smallest entry 1 which corresponds to the edge (v_5, v_2). By examining all entries in v_2- and v_5-rows, we find that the vertex other than v_2 and v_5 which corresponds to the smallest entry is v_3.

Thus, v_3 is closest to the edge (v_2, v_5). Let us connect v_3 to the edge (v_2, v_5).

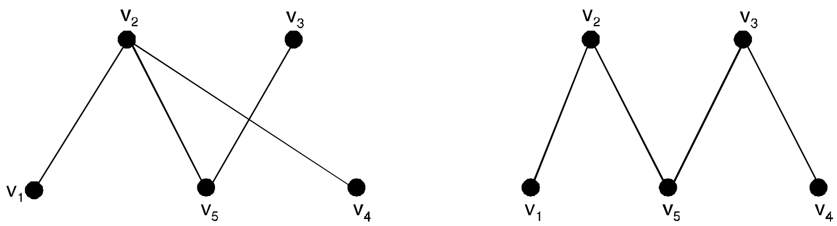
Thus, the edges (v_1, v_2), (v_2, v_5), (v_5, v_3) belong to a minimal spanning tree. The vertex left over at this stage is v_4 which is joined to v_2, v_3 and v_5 in the given graph.

Among the edges that contain v_4, the edges (v_2, v_4) and (v_3, v_4) have minimum weights.

Therefore, we can include either of these edges in the minimal spanning tree.

Accordingly, the edges (v_1, v_2), (v_2, v_5), (v_5, v_3) together with the edge (v_2, v_4) or the edge (v_3, v_4) constitute a minimal spanning tree.

Thus, for the given graph, there are two minimal spanning trees as shown below :



(a) (b)

Fig. 4.41.

Problem 4.23. *Use Prim's algorithm to find an optimal tree for the graph in Fig. (4.42).*

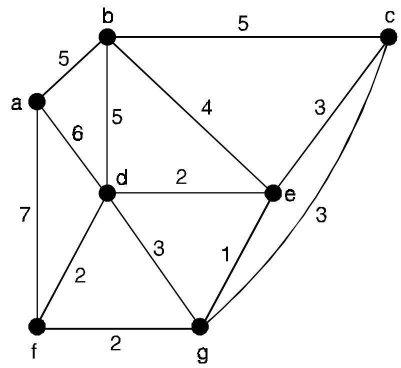


Fig. 4.42.

Solution. Prim's algorithm generates an optimal tree as follows :

Initialization : $i = 1$, P = {a}, N = {b, c, d, e, f, g}, T = ϕ.

First iteration : T = {{a, b)}, P = {a, b}, N = {c, d, e, f, g}, $i = 2$.

Second iteration : T = {{a, b}, {b, e}}, P = {a, b, e}, N = {c, d, f, g}, $i = 3$

Third iteration : T = {{a, b}, {b, e}, {e, g}}, P = {a, b, e, g}, N = {c, d, f}, $i = 4$

Fourth iteration : T = {{a, b}, {b, e}, {e, g}, {d, e}}, P = {a, b, e, g, d}, N = {c, f}, $i = 5$.

Fifth iteration : T = {{a, b}, {b, e}, {e, g}, {d, e}, {f, g}},
P = {a, b, e, g, d, f}, N = {c}, $i = 6$.

Sixth iteration : T = {{a, b}, {b, e}, {e, g}, {d, e}, {f, g}, {c, g}},
P = {a, b, e, g, d, f, c} = V, $n = \phi$, $i = 7 = |\,V\,|$.

Hence T is an optimal spanning tree of weight 17 for G, as seen in Fig. (4.43).

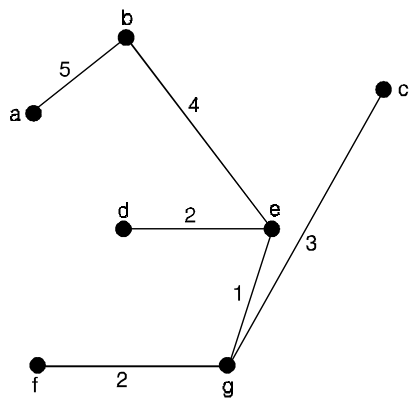


Fig. 4.43.

Note that the minimal spanning tree obtained here differs from that in Fig. (4.44). So this type of spanning tree need not be unique.

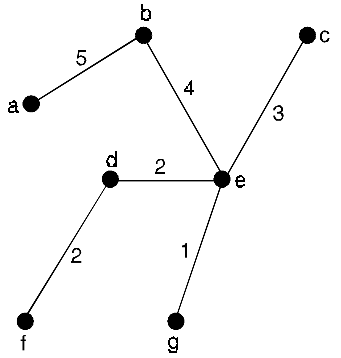


Fig. 4.44.

Problem 4.24. *Using Prim's algorithm, find a minimal spanning tree for the weighted graph shown in Fig. (4.45).*

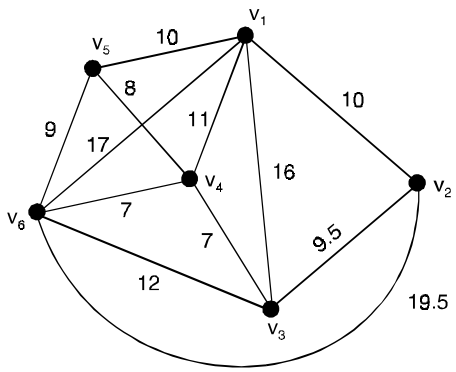


Fig. 4.45.

Solution. Let us prepare the following table indicating the weights of the edges joining every pair of vertices

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	–	10	16	11	10	17
v_2	10	–	9.5	∞	∞	19.5
v_3	16	9.5	–	7	∞	12
v_4	11	∞	7	–	8	7
v_5	10	∞	∞	8	–	9
v_6	17	19.5	12	7	9	–

Now we start with the first row (v_1-row) and pick the smallest entry therein. This is 10 which corresponds to the edge (v_1, v_2) or (v_1, v_5).

Let us select one of the these edges, say (v_1, v_5). Now, by examining all the entries in v_1- and v_5-rows, we find that the vertex other than v_1 and v_5 which corresponds to the smallest entry is v_4.

Thus, v_4 is closest to the edge (v_1, v_5). Let us connect v_4 to the edge (v_1, v_5).

Let us now examine the v_4-row and note that the smallest entry therein is 7 which corresponds to the edge (v_4, v_3) or (v_4, v_6).

Let us choose one of these, say (v_4, v_6). By examining the entries in v_4- and v_6-rows, we find that the vertex other than v_4 and v_6 which corresponds to the smallest entry is v_3.

Thus, v_3 is closest to the edge (v_4, v_6). Let us connect v_3 to the edge (v_4, v_6).

Let us examine the v_3-row and pick the edge (v_3, v_4) which corresponds to the smallest entry.

By examining the entries in v_3- and v_4-rows, we find that the vertex other than v_3 and v_4 which corresponds to the smallest entry is v_2.

Thus, v_2 is closest to the edge (v_3, v_4). Let us connect v_2 to (v_3, v_4).

We stop the process, because we have now connected all the six vertices with five edges, (v_1, v_5), (v_5, v_4), (v_4, v_6), (v_4, v_3), (v_3, v_2).

The corresponding graph is as shown below. This graph constitutes a minimal spanning tree of the given graph. We note that its weight is 41.5 units.

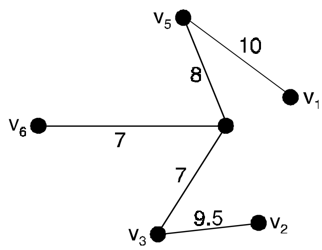


Fig. 4.46.

4.3.6. The labeling algorithm

Step 1 : Let N_1 be the set of all nodes connected in the source by an edge with positive excess capacity. Label each j in N_1 with [E_j, 1], where E_j is the excess capacity e_{1j} of edge (1, j). The 1 in the label indicates that j is connected to the source, node 1.

Step 2 : Let node j in N_1 be the node with smallest node number and let N_2 (j) be the set of all unlabeled nodes, other than the source, that are joined to node j and have positive excess capacity.

Suppose that node k is in N_2 (j) and (j, k) is the edge with positive excess capactiy.

Label node k with [E_k, j], where E_k is the minimum of E_j and the excess capacity e_{jk} of edge (j, k).

When all the nodes in N_2 (j) are labeled in this way, repeat this process for the other nodes in N_1.

Let $N_2 = \bigcup_{j \in N_1} N_2(j)$.

Note that after step 1, we have labeled each node j in N_1 with E_j, the amount of material that can flow from the source to j through one edge and with the information that this flow came from node 1.

In step 2, previously unlabeled nodes k that can be reached from the source by a path $\pi : 1, j, k$ are labeled with $[E_{k, j}]$.

Here E_k is the maximum flow that can pass through π since it is the smaller of the amount that can reach j and the amount that can then pass on to k.

Thus, when step 2, is finished, we have constructed two-step paths to all nodes in N_2.

The label for each of these nodes records the total flow that can reach the node through the path and its immediate predecessor in the path.

We attempt to continue this construction increasing the lengths of the paths until we reach the sink (if possible).

Then the total flow can be increased and we can retrace the path used for this increase.

Step 3 : Repeat step 2, labelling all previously unlabeled nodes N_3 that can be reached from a node in N_2 by an edge having positive excess capacity.

Continue this process forming sets N_4, N_5, ... until after a finite number of steps either

(*i*) The sink has not been labeled and no other nodes can be labeled. It can happen that no nodes have been labeled, remember that the source is not labeled. or

(*ii*) The sink has been labeled.

Step 4 : In case (*i*), the algorithm terminates and the total flow then is a maximum flow.

Step 5 : In case (*ii*) the sink, node n, has been labeled with $[E_n, m]$ where E_n is the amount of extra flow that can be made to reach the sink through a path π. We examine π in reverse order. If each $(i, j) \in N$, then we increase the flow in (i, j) by E_n and decrease the excess capacity e_{ij} by the same amount.

Simultaneously, we increase the excess capacity of the (virtual) edge (j, i) by E_n since there is that much more flow in (i, j) to reverse.

If on the other hand, $(i, j) \notin N$, we decrease the flow in (j, i) by E_n and increase its excess capacity by E_n.

We simultaneously decrease the excess capacity in (i, j) by the same amount, since there is less flow in (i, j) to reverse.

We now have a new flow that is E_n units greater than before and we return to step 1.

Problem 4.25. *Use the labeling algorithm to find a maximum flow for the network in Fig. (4.47).*

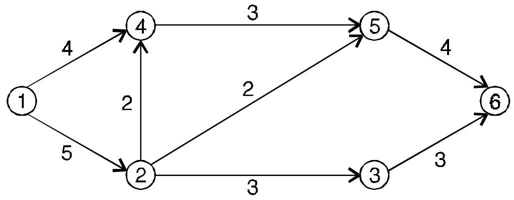


Fig. 4.47.

Solution. Fig. (4.48) shows the network with initial capacities of all edges in G. The initial flow in all edges is zero.

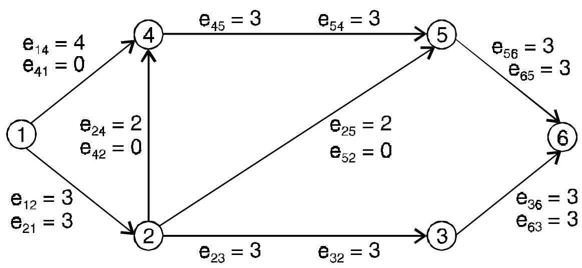


Fig. 4.48.

Step 1 : Starting at the source, we can reach nodes 2 and 4 by edges having excess capacity, so $N_1 = \{2, 4\}$.

We label nodes 2 and 4 with the labels [5, 1] and [4, 1], respectively, as shown in Fig. (4.49).

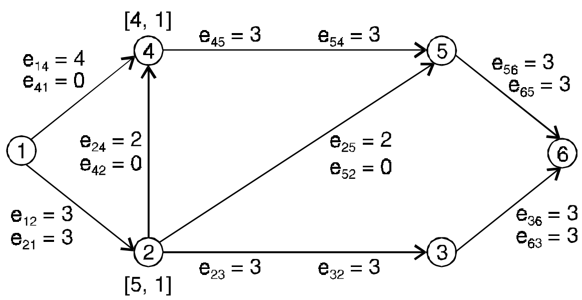


Fig. 4.49.

Step 2 : From node 2 we can reach nodes 5 and 3 using edges with positive excess capacity.

Node 5 is labeled with [2, 2] since only two additional units of flow can pass through edge (2, 5).

Node 3 is labeled with [3, 2] since only 3 additional units of flow can pass through edge (2, 3). The result of this step is shown in Fig. (4.50).

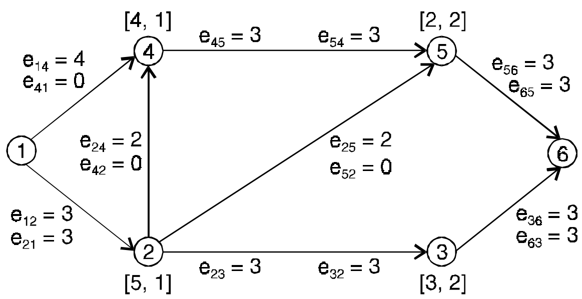


Fig. 4.50.

We cannot travel from node 4 to any unlabeled node by one edge. Thus, $N_2 = \{3, 5\}$ and step 2 is complete.

Step 3 : We repeat step 2 using N_2. We can reach the sink from node 3 and 3 units through edge (3, 6). Thus, the sink is labeled with [3, 3].

Step 4 : We work backward through the path 1, 2, 3, 6 and subtract 3 from the excess capacity of each edge, indicating an increased flow through that edge, and adding an equal amount to the excess capacities of the (virtual) edges. We now return to step 1 with the situation shown in Fig. (4.51).

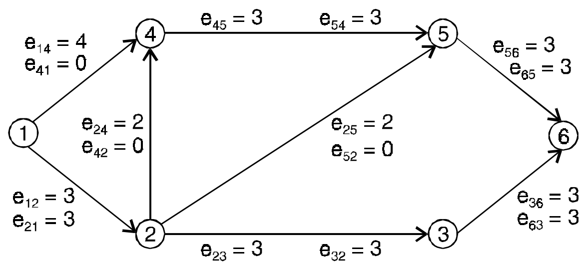


Fig. 4.51.

Proceeding as before, nodes 2 and 4 are labeled [2, 1] and [4, 1] respectively.

Note that E_2 is now only 2 units, the new excess capacity of edge (1, 2).

Node 2 can no longer be used to label node 3, since there is no excess capacity in the edge (2, 3). But node 5 now will be labeled [2, 2]. Once again no unlabled node can be reached from node 4, so we move to step 3.

Here we can reach node 6 from node 5 so node 6 is labeled with [2, 5].

The final result of step 3 is shown in Fig. (4.52), and we have increased the flow by 2 units to a total of 5 units.

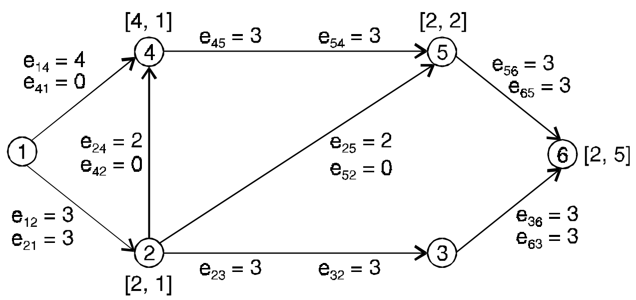


Fig. 4.52.

We move to step 4 again and work back along the path 1, 2, 5, 6, subtracting 2 from the excess capacities of these edges and adding 2 to the capacities of the corresponding (virtual) edges.

We return to step 1 with Fig. (4.53).

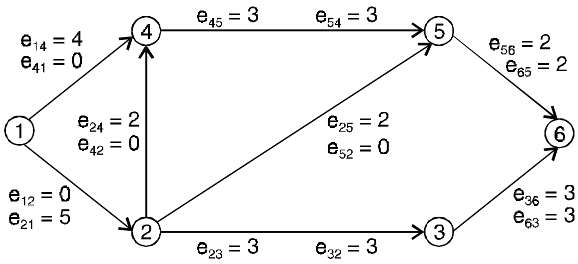


Fig. 4.53.

This time steps 1 and 2 produce the following results. Only node 4 is labeled from node 1, with [4, 1]. Node 5 is the only node labeled from node 4, with [3, 4] step 3 begins with Fig. (4.54).

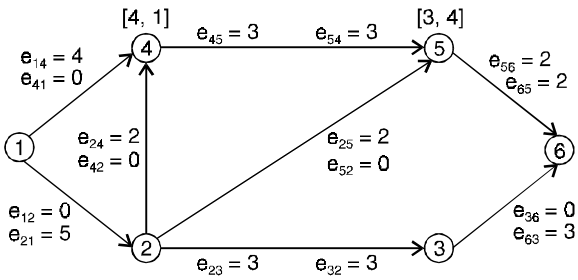


Fig. 4.54.

At this point, node 5 could label node 2 using the excess capacity of edge (5, 2).

However, node 5 can also be used to label the sink. The sink is labeled [2, 5] and the total flow is increased to 7 units. In step 5, we work back along the path 1, 4, 5, 6, adjusting excess capacities. We return to step 1 with the configuration shown in Fig. (4.55).

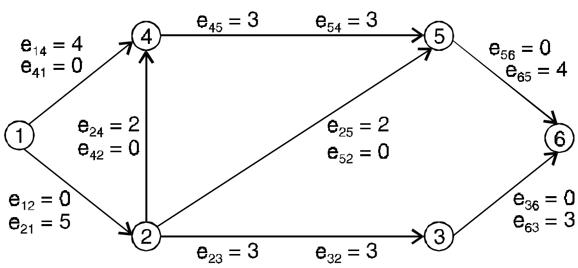


Fig. 4.55.

Verify that after steps 1, 2 and 3, nodes 4, 5 and 2 have been labeled as shown in Fig. (4.56) and no further labeling is possible. The final labeling of node 2 uses the virtual edge (5, 2).

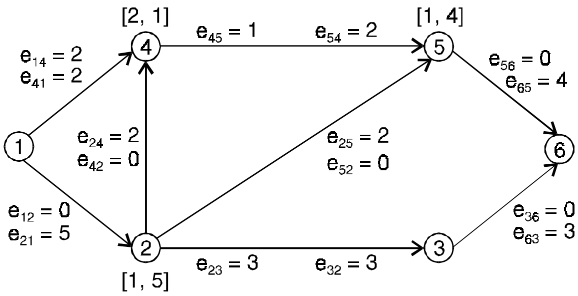


Fig. 4.56.

Thus, the final overall flow has value 7. By subtracting the final excess capacity e_{ij} of each edge (i, j) in N from the capacity C_{ij}, the flow F that produces the maximum value 7 can be see in Fig. (4.57).

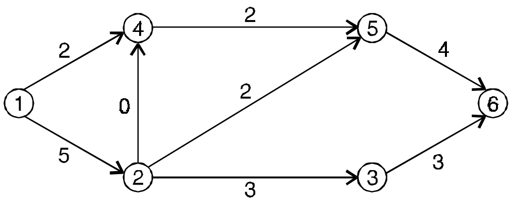


Fig. 4.57.

4.3.7. Reachability

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

For any subset U of the vertex set V, the reachable set of U is the set of all vertices which are reachable from any vertex set of S and this set is denoted by R(S).

For example, in the graph given below :

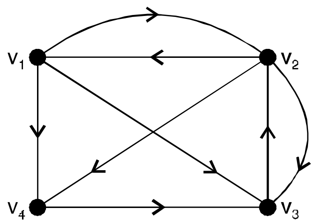


Fig. 4.58.

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

4.3.8. Distance and diameter

In a connected graph G, the distance between the vertices u and v, denoted by $d(u, v)$ is the length of the shortest path.

In Fig. (4.59)(a), $d(a, f) = 2$ and in Fig. (4.59) (b), $d(a, e) = 3$.

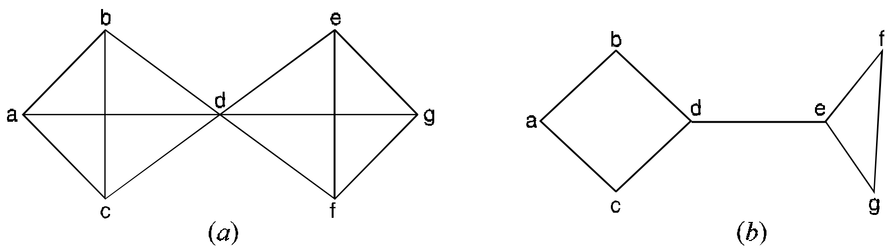


Fig. 4.59.

The distance function as defined above has the following properties.

If u, v and w are any three verties of a connected graph then

(i) $d(u, v) \geq 0$ and $d(u, v) = 0$ if $u = v$.

(ii) $d(u, v) = d(v, u)$ and

(iii) $d(u, v) \geq d(u, w) + d(w, v)$

This shows that distance in a graph is metric.

The diameter of G, written as diam (G) is the maximum distance between any two vertices in G. In Fig. (4.59) (a), diam (G) = 2 and in Fig. (4.59) (b), diam (G) = 4.

4.3.9. 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.**

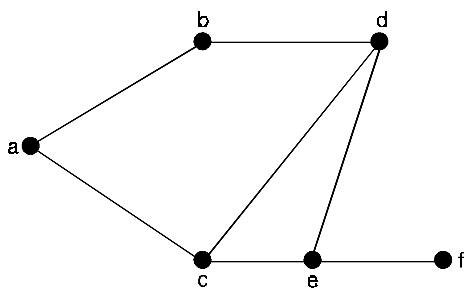
Analogously, an edge whose removal produces a graph with more connected components then the original graph is called a **cut edge** or **bridge.**

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.



Fig, 4.60.

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.

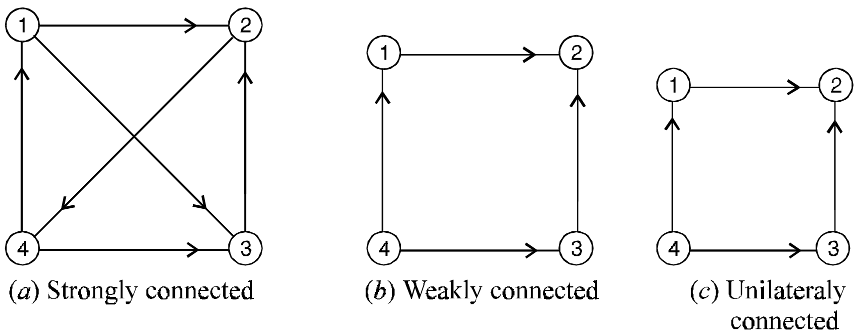


Fig. 4.61. Connectivity in digraphs.

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 λ(G) or E(G).

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

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

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 λ(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.

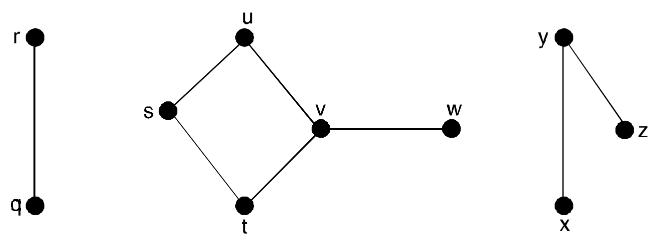


Fig. 4.62.

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 ?*

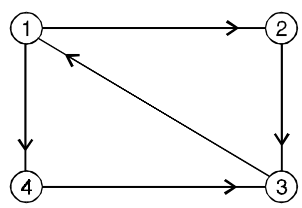


Fig. 4.63.

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 4.6. *The edge connectivity of a graph G cannot exceed the minimum degree of a vertex in G, i.e.,* $\lambda(G) \leq \delta(G)$.

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.

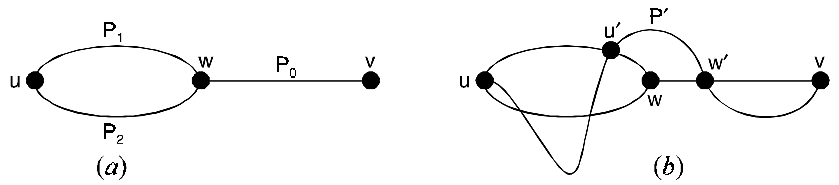


Fig. 4.64. Pahs in blocks.

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.

Problem 4.28. *Find the V(G), E(G) and deg (G) for the graph of the Figure (4.65).*

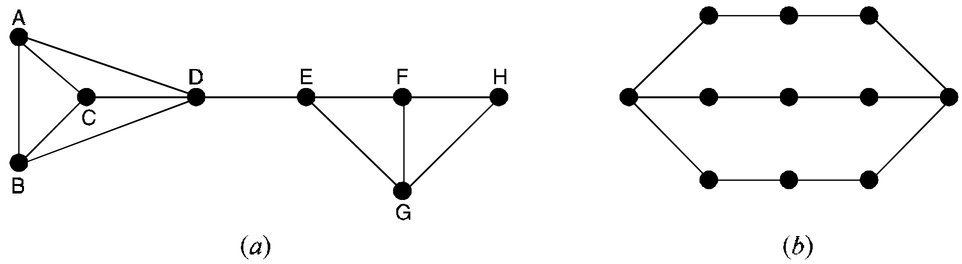


Fig. 4.65

Solution. (*a*) The degree of the graph G, deg (G) = 5.

If we remove node D from the graph then graph becomes two components graph.

Thus, V(G) = 1.

By the removal of arcs (D, H) and (D, E) the graph G turns into two components graph.

Hence E(G) = 2.

(*b*) Here deg(G) = 3

V(G) = 2 and E(G) = 2.

Problem 4.29. *Find the E(G) and V(G) of the graph shown in Figure (4.66).*

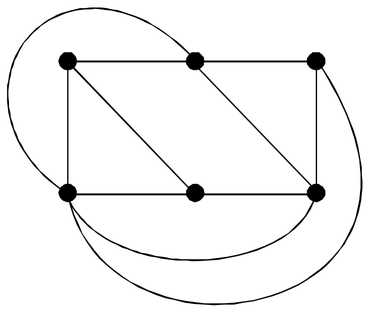


Fig. 4.66.

Solution. To calculate number of arc disjoint paths between any pair of nodes, maximum flow between that pair of node is calculated.

The procedure is shown in the given table. It is assumed that :

(*i*) an arc can carry only one unit of flow and

(*ii*) a node has infinite capacity.

S.No.	*Node Pair*	*Maximum Flow*	*Remark*
1.	(1, 2)	3	Three arcs from node 1 can carry at the most 3 units of flow and node 2 can receive all of them.
2.	(1, 3)	3	same as above
3.	(1, 4)	3	same as above
4.	(1, 5)	3	same as above
5.	(1, 6)	3	same as above
6.	(2, 3)	3	Though node 2 can send 4 units of flow, node 3 can accept only 3 units.
7.	(2, 4)	4	Node 2 can sent 4 units and node 4 can accept all of them.
8.	(2, 5)	3	same as in sl. no. 6
9.	(2, 6)	4	same as in sl. no. 7
10.	(3, 4)	3	same as in sl. no. 1
11.	(3, 5)	3	same as in sl. no. 1

12.	(3, 6)	3	same as in sl. no. 1
13.	(4, 5)	3	same as in sl. no. 6
14.	(4, 6)	4	same as in sl. no. 7
15.	(5, 6)	3	same as in sl. no. 1

The minimum value of maximum flow between any pair of node 3. This is the count of minimum number of arc disjoint path between any pair of nodes in G.

Hence E(G) = 3.

Similarly, we can compute V(G) of the graph.

The following assumptions are made to compute node disjoint path between one node to another :

(*i*) Arc has infinite capacity so it can carry any amount of flow.

(*ii*) Any intermediate node in the path can accept I units of flow along any one incoming arc and can pass only one unit at a time along any one outgoing arc. If an intermediate node *b* has 5 incoming arcs from a node *a* then *b* can accept only one unit of flow from *a*.

Similarly if *b* has 4 outgoing arcs, it can pass only one unit of flow along any one out of four arcs.

(*iii*) If nodes are adjacent then it can sustain loss of all other nodes, so maximum flow is assumed to be $n - 1$, where n is $|V|$.

The calculation is shown in the given table. From the table, it is clear that V(G) = 3.

S.No.	*Node Pair*	*Maximum Flow*	*Remark*
1.	(1, 2)	$n - 1$	Both nodes 1 and 2 are adjacent.
2.	(1, 3)	3	Node disjoint paths are (1, 2, 3), (1, 5, 4, 3) and (1, 6, 3)
3.	(1, 4)	3	Node disjoint paths are (1, 2, 4), (1, 5, 4) and (1, 6, 4)
4.	(1, 5)	$n - 1$	Both nodes 1 and 5 are adjacent
5.	(1, 6)	$n - 1$	Both nodes 1 and 6 are adjacent
6.	(2, 3)	$n - 1$	Both nodes 2 and 3 are adjacent
7.	(2, 4)	$n - 1$	Both nodes 2 and 4 are adjacent
8.	(2, 5)	3	Noded disjoint paths are : (2, 1, 5), (2, 4, 5) and (2, 6, 5)
9.	(2, 6)	$n - 1$	Both nodes 2 and 6 are adjacent
10.	(3, 4)	$n - 1$	Both nodes 3 and 4 are adjacent
11.	(3, 5)	3	Noded disjoint paths are : (3, 4, 5), (3, 2, 1, 5) and (3, 6, 5)
12.	(3, 6)	$n - 1$	Both nodes 1 and 2 are adjacent
13.	(4, 5)	$n - 1$	Both nodes 1 and 2 are adjacent
14.	(4, 6)	$n - 1$	Both nodes 1 and 2 are adjacent
15.	(5, 6)	$n - 1$	Both nodes 1 and 2 are adjacent

Theorem 4.12. *In any graph G, V(G) ≤ E(G) ≤ deg (G).*

Proof. Let deg(G) = *n*.

Then there exists a node V in G such that degree of V is *n.*

If we drop all arcs for which V is an incidence (a node is called an incidence of an arc if the node is either a start or an end point of the arc), the graph becomes disconnected.

Thus, E(G) cannot exceed *n* otherwise there exists a node which is incidence of $m > n$ number of arcs. That is in contradiction with the assumption that

deg(G) = *n*

Thus, E(G) ≤ deg(G) ...(1)

Next, Let E(G) = *r*.

Then there exist a pair of nodes such that there are *r* disjoint paths between them.

These *r* paths may cross through S ≤ *r* number of nodes.

If we remove these *s* nodes from the graph, the *r* arcs get deleted from the graph making the graph a disconnected.

That means V(G) cannot exceed *r*.

Thus V(G) ≤ E(G) ...(2)

Combining results (1) and (2), we have

V(G) ≤ E(G) ≤ deg(G).

Theorem 4.13. *Let v is a cut point of a connected graph G = (V, E). The remaining set of vertex V–{v} can be partitioned into two non empty disjoint subsets U and W such that for any node u ∈ U and w ∈ W, the node v lies on every u–w path.*

Proof. When cut point *v* is removed from G it becomes disconnected.

Let U be a set of vertices of the largest connected subgraph of G and W = V – {*v*} – U.

Let *v* is not on every *u*–*w* path.

This implies that a path from *u* to *w* exists even after removal of *v* from G.

That means U is not the set of vertices of largest connected subgraph of G after removal of *v*.

This is a contrary to the assumption that U is the largest component.

Hence *v* lies on every *u*–*w* path.

4.4 TRANSPORT NETWORKS

Let N = (V, E) be a loop-free connected directed graph. Then N is called a **network,** or **transport network,** if the following conditions are satisfied :

(*i*) There exists a unique vertex *a* ∈ V with *id*(*a*), the in degree of *a*, equal to O. This vertex *a* is called the **source.**

(*ii*) There is a unique vertex *z* ∈ V, called the **sink,** where *od*(*z*), the out degree of *z*, equals O.

(*iii*) The graph N is weighted, so there is a function from E to the set of non negative integers that assigns to each edge *e* = (*v*, *w*) ∈ E a capacity, denoted by *c*(*e*) = *c*(*v*, *w*).

If N = (V, E) is a transport network, a function *f* from E to the non negative integers is called a **flow** for N if

(i) $f(e) \le c(e)$ for each edge $e \in$ E, and

(ii) for each $v \in$ V, other than the source a or the sink z, $\sum_{w \in V} f(w, v) = \sum_{w \in V} f(v, w)$

If there is no edge (v, w), then $f(v, w) = 0$.

Let f be a flow for a transport network N = (V, E)

(i) An edge e of the network is called **saturated** if $f(e) = c(e)$. When $f(e) < c(e)$, the edge is called **unsaturated.**

(ii) If a is the source of N, then val(f) = $\sum_{v \in V} f(a, v)$ is called the value of the **flow.**

If N = (V, E) is a transport network and C is a cut-set for the undirected graph associated with N, then C is called a **cut,** or an ***a–z*** **cut,** if the removal of the edges in C from the network results in the separation of a and z.

For example, the graph in Fig. (4.67) is a transport network. Here vertex a is the source, the sink is at vertex z, and capacities are shown beside each edge. Since $c(a, b) + c(a, g) = 5 + 7 = 12$, the amount of the commodity being transported from a to z cannot exceed 12. With $c(d, z) + c(h, z) = 5 + 6 = 11$, the amount is further restricted to be no greater than 11.

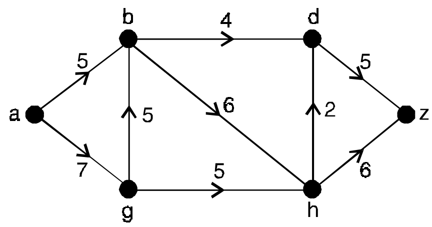


Fig. 4.67.

For the network in Fig. (4.68), the label x, y on each edge e is determined so that $x = c(e)$ and y is the value assigned for a possible flow f. The label on each edge e satisfies $f(e) \le c(e)$.

In part (a) of the Fig. (4.68), the flow into vertex g is 5, but the flow out from that vertex is 2 + 2 = 4.

Hence the function f is not a flow in this case.

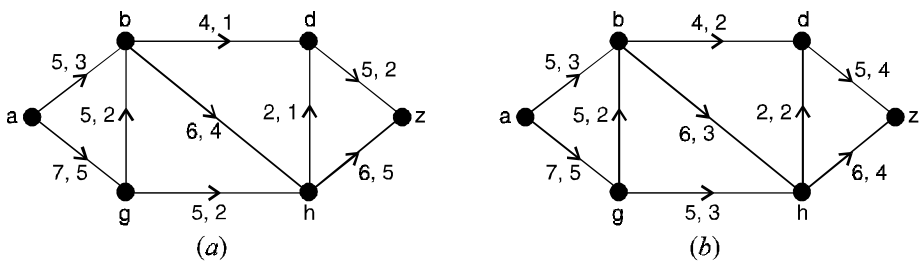


Fig. 4.68.

For the network in Fig. (4.68) (*b*), only the edge (*h*, *d*) is saturated. All other edges are unsaturated.

The value of the flow in this network is

$$\text{val}(f) = \sum_{v \in V} f(a, v) = f(a, b) + f(a, g) = 3 + 5 = 8.$$

We observe that in the network of Fig. (4.68) (*b*)

$$\sum_{v \in V} f(a, v) = 3 + 5 = 8 = 4 + 4 = f(d, z) + f(h, z) = \sum_{v \in V} f(v, z).$$

Consequently, the total flow leaving the source a equals the total flow into the sink *z*.

Fig. (4.69) indicates a cut for the given network (dotted curves). The cut C, consists of the undirected edges {*a*, *g*}, {*b*, *d*}, {*b*, *g*} and {*b*, *h*}. This cut partitions the vertices of the network into the two sets P = {*a*, *b*} and its complement $\overline{P}$ = {*d*, *g*, *h*, *z*}, so C_1 is denoted as (P, $\overline{P}$).

The capacity of a cut, denoted C(P, $\overline{P}$), is defined by $C(P, \overline{P}) = \sum_{\substack{v \in P \\ w \in \overline{P}}} C(v, w)$, the sum of the capacities of all edges (*v*, *w*), where $v \in$ P and $w \in \overline{P}$.

In this example, C(P, $\overline{P}$) = *c*(*a*, *g*) + *c*(*b*, *d*) + *c*(*b*, *h*) = 7 + 4 + 6 = 17.

The cut c_2 induces the vertex partition Q = {*a*, *b*, *g*}.

$\overline{Q}$ = {*d*, *h*, *z*} and has capacity *c*(Q, $\overline{Q}$) = *c*(*b*, *d*) + *c*(*b*, *h*) + *c*(*g*, *h*) = 4 + 6 + 5 = 15.

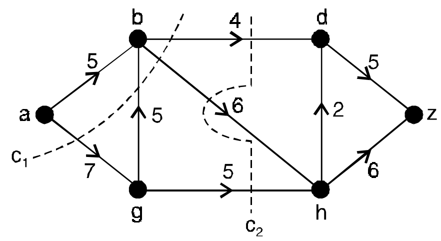


Fig. 4.69.

Theorem 4.14. *Let f be a flow in a network N = (V, E). If C = (P, $\overline{P}$) is any cut in N, then val(f) cannot exceed c(P, $\overline{P}$).*

Proof. Let vertex *a* be the source in N and vertex *z* the sink. Since *id*(*a*) = 0, it follows that for all $w \in$ V, *f*(*w*, *a*) = 0.

Consequently, $\text{val}(f) = \sum_{v \in V} f(a, v) = \sum_{v \in V} f(a, v) - \sum_{w \in V} f(w, a)$

By the definition of a flow, for all $x \in$ P, $x \neq a$,

$$\sum_{v \in V} f(x, v) - \sum_{w \in V} f(w, x) = 0.$$

Adding the results in the above equations yields

$$\text{val}(f) = \left[\sum_{v \in V} f(a, v) - \sum_{w \in V} f(w, a)\right] + \sum_{\substack{w \in P \\ x \neq a}} \left[\sum_{v \in V} f(x, v) - \sum_{w \in V} f(w, x)\right]$$

$$= \sum_{\substack{x \in P \\ v \in V}} f(x, v) - \sum_{\substack{x \in P \\ w \in V}} f(w, x)$$

$$= \left[\sum_{\substack{x \in P \\ v \in P}} f(x, v) + \sum_{\substack{x \in P \\ v \in \overline{P}}} f(x, v)\right] - \left[\sum_{\substack{x \in P \\ w \in P}} f(w, x) + \sum_{\substack{x \in P \\ w \in \overline{P}}} f(w, x)\right]$$

Since $\sum_{\substack{x \in P \\ v \in P}} f(x, v)$ and $\sum_{\substack{x \in P \\ w \in P}} f(w, x)$ are summed over the same set of all ordered pairs in P × P, these summations are equal.

Consequently, $\text{val}(f) = \sum_{\substack{x \in P \\ v \in \overline{P}}} f(x, v) - \sum_{\substack{x \in P \\ w \in \overline{P}}} f(w, x)$

For all x, $w \in v$, $f(w, x) \geq 0$, so

$$\sum_{\substack{x \in P \\ w \in \overline{P}}} f(w, x) \geq 0 \text{ and } \text{val}(f) \leq \sum_{\substack{x \in P \\ v \in \overline{P}}} f(x, v) \leq \sum_{\substack{x \in P \\ v \in \overline{P}}} c(x, v) = c(P, \overline{P}).$$

Corollary :

If f is a flow in a transport network N = (V, E), then the value of the flow from the source a is equal to the value of the flow into the sink z.

Proof. Let P = $\{a\}$, $\overline{P}$ = V – $\{a\}$, and Q = V – $\{z\}$.

From the above observation,

$$\sum_{\substack{x \in P \\ v \in \overline{P}}} f(x, v) - \sum_{\substack{x \in P \\ w \in \overline{P}}} f(w, x) = \text{val}(f) = \sum_{\substack{y \in Q \\ v \in \overline{Q}}} f(y, v) - \sum_{\substack{y \in Q \\ w \in \overline{Q}}} f(w, y)$$

With P = $\{a\}$ and $id(a) = 0$, we find that

$$\sum_{\substack{x \in P \\ w \in \overline{P}}} f(w, x) = \sum_{w \in \overline{P}} f(w, a) = 0.$$

Similarly, for $\overline{Q} = \{z\}$ and $od(z) = 0$, it follows that

$$\sum_{\substack{y \in Q \\ w \in \overline{Q}}} f(w, y) = \sum_{y \in Q} f(z, y) = 0$$

Consequently, $\displaystyle\sum_{\substack{x \in P \\ v \in \overline{P}}} f(x, v) = \sum_{v \in \overline{P}} f(a, v) = \text{val}(f)$

$$= \sum_{\substack{y \in Q \\ v \in \overline{Q}}} f(y, v) = \sum_{y \in Q} f(y, z).$$

Theorem 4.15. *The value of any flow in a given transport network is less than or equal to the capacity of any cut in the network.*

Proof. Let ϕ be a flow and $(P, \overline{P})$ be a cut in a transport network. For the source a,

$$\sum_{\text{all } i} \phi(a, i) - \sum_{\text{all } j} \phi(j, a) = \sum_{\text{all } i} \phi(a, i) = \phi v \qquad ...(1)$$

Since $\phi(j, a) = 0$ for any j. For a vertex P other than a in P,

$$\sum_{\text{all } i} \phi(P, i) - \sum_{\text{all } j} \phi(j, P) = 0 \qquad ...(2)$$

Combining (1) and (2), we have

$$\phi_v = \sum_{p \in P} \left[\sum_{\text{all } i} \phi(P, i) - \sum_{\text{all } j} \phi(j, P) \right]$$

$$= \sum_{p \in P\,;\,\text{all } i} \phi(P, i) - \sum_{p \in P\,;\,\text{all } j} \phi(j, P)$$

$$= \sum_{p \in P\,;\,i \in P} \phi(P, i) + \sum_{p \in P\,;\,i \in \overline{P}} \phi(P, i) - \left[\sum_{p \in P\,;\,j \in P} \phi(j, P) + \sum_{p \in P\,;\,j \in \overline{P}} \phi(j, P) \right] \qquad ...(3)$$

Note that $\displaystyle\sum_{p \in P\,;\,i \in P} \phi(P, i) = \sum_{p \in P\,;\,j \in P} \phi(j, P)$

because both sums run through all the vertices in P. Thus, (3) becomes

$$\phi_v = \sum_{p \in P\,;\,i \in \overline{P}} \phi(P, i) - \sum_{p \in P\,;\,j \in \overline{P}} \phi(j, P) \qquad ...(4)$$

But, sine $\displaystyle\sum_{p \in P\,;\,j \in \overline{P}} \phi(j, P)$ is always a non-negative quantity.

We have

$$\phi_v \leqq \sum_{p \in P;\, i \in \bar{P}} \phi(P, i) \leqq \sum_{p \in P;\, i \in \bar{P}} w(P, i) = w(P, \bar{P}).$$

Theorem 4.16. *In any directed network, the value of an (s, t)–flow never exceeds the capacity of any (s, t)–cut.*

Proof. Let $F = \{f_{uv}\}$ be any (s, t)–flow and $\{S, T\}$ any (s, t)–cut.

Conservation of flow tells us that $\sum_v f_{uv} - \sum_v f_{vu} = 0$

for any $u \in s$, $u \neq s$. (the possibility $u = t$ is excluded because $t \notin S$)

Hence, $\text{val}(F) = \sum_{v \in V} f_{sv} - \sum_{v \in V} f_{vs}$

$$= \sum_{u \in S} \left(\sum_{v \in V} f_{uv} - \sum_{v \in V} f_{vu} \right)$$

(since the term in parentheses is 0 except for $u = s$)

$$= \sum_{u \in S,\, v \in V} f_{uv} - \sum_{u \in S,\, v \in V} f_{vu}.$$

Since {S, T} is a partition, this last sum can be written

$$\sum_{u \in S,\, v \in S} f_{uv} + \sum_{u \in S,\, v \in T} f_{uv} - \sum_{u \in S,\, v \in S} f_{vu} - \sum_{u \in S,\, v \in T} f_{vu}$$

$$= \sum_{u \in S,\, v \in S} f_{uv} - \sum_{u \in S,\, v \in S} f_{vu} + \sum_{u \in S,\, v \in T} (f_{uv} - f_{vu})$$

The first two terms in the line are the same, so we obtain

$$\text{val}(F) = \sum_{u \in S,\, v \in T} (f_{uv} - f_{vu}).$$

But $f_{uv} \leq C_{uv}$ and $f_{vu} \geq 0$, so $f_{uv} - f_{vu} \leq C_{uv}$ for all u and v.

Therefore, $\text{val}(F) \leq \sum_{u \in S,\, v \in T} C_{uv} = \text{cap}(S, T)$ as desired.

Corollary 1.

If F is any (s, t)–flow and (S, T) is any (s, t)–cut, then val(F), $\sum_{u \in S,\, v \in T} (f_{uv} - f_{vu})$.

With refernece to the network in Fig. (4.70) and the cut $S = \{s, a, c\}$, $T = \{b, d, t\}$, the sum specified in the corollary is

$$\sum_{u \in S,\, v \in T} (f_{uv} - f_{vu}) = f_{sb} + f_{ad} - f_{bc} + f_{ct} = 2 + 0 - 1 + 11 = 12,$$

which is the value of the flow in this network.

Corollary 2.

Suppose there exists some (s, t)–flow F and some (s, t)–cut {S, T} such that the value of F equals the capacity of {S, T}. Then val(F) is the maximum value of any flow and cap(S, T) is the minimum capacity of any cut.

Proof. Let F_1 be any flow. To see that $val(F_1) \le val(F)$, note that the theorem says that $val(F_1) \le cap(S, T)$ and, by hypothesis, cap(S, T) = val(F).

So val(F) is maximum.

In any directed network, there is always a flow and a cut such that the value of the flow is the capacity of the cut, such a flow has maximum value.

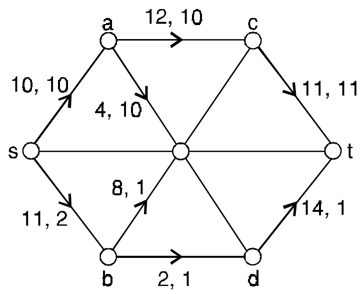


Fig. 4.70. (*s*, *t*)-flow.

4.5 MAX–FLOW MIN–CUT THEOREM

In any network, the value of any maximum flow is equal to the capacity of any minimum cut.

First proof :

Suppose first that the capacity of each arc is an integer. Then the network can be regarded as a digraph D whose capacities represent the number of arcs connecting the various vertices (as in Figs. (4.71) and (4.72)).

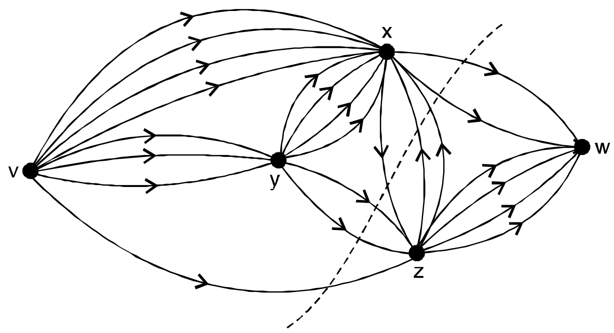


Fig. 4.71.

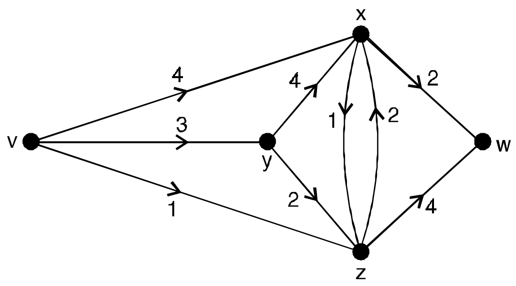


Fig. 4.72

The value of a maximum flow is the total number of arc-disjoint paths from *v* to *w* in D, and the capacity of a minimum cut is the minimum number of arcs in a *vw*-disconnecting set of D.

The extension of this result to networks in which the capacities are rational numbers is effected by multiplying these capacities by a suitable integer *d* to make them integers.

We then have the case described above, and the result follows on dividing by *d*.

Finally, if some capacities are irrational, then we approximate them as closely as we please by rationals and use the above result.

By choosing these rationals carefully, we can ensure that the value of any maximum flow and the capacity of any minimum cut are altered by an amount that is as small as we wish.

Note that, in practical examples, irrational capacities rarely occur, since the capacities are usually given in decimal form.

Second Proof

Since the value of any maximum flow cannot exceed the capacity of any minimum cut, it is sufficient to prove the existence of a cut whose capacity is equal to the value of a given maximum flow.

Let ϕ be a maximum flow. We define two sets V and W of vertices of the network as follows.

If G is the underlying graph of the network, then a vertex *z* is contained in V, if and only if there exists in G a path $v = v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow \ldots\ldots \rightarrow v_{m-1} \rightarrow v_m = z$, such that each edge v_iv_{i+1} corresponds either to an unsaturated arc v_iv_{i+1}, or to an arc $v_{i+1}v_i$ that carries a non-zero flow. The set W consists of all those vertices that do not lie in V.

For example, in Fig. (4.73), the set V consists of the vertices *v*, *x* and *y*, and the set W consists of the vertices *z* and *w*.

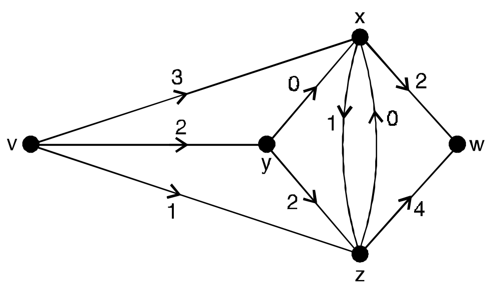


Fig. 4.73.

Clearly, v is contained in V. We now show that W contains the vertex w.

If this is not so, then w lies in V, and hence there exists in G a path $v \to v_1 \to v_2 \to \ldots\ldots \to v_{m-1} \to w$ of the above type.

We now choose a positive number ε that does not exceed the amount needed to saturate any unsaturated arc $v_i v_{i+1}$, and does not exceed the flow in any arc $v_{i+1} v_i$ that carries a non-zero flow.

It is now easy to see that, if we increase by ε the flow in all arcs of the first type and decrease by ε the flow in all arcs of the second type, then we increase the value of the flow by ε, contradicting our assumption that ϕ is a maximum flow.

It follows that w lies in W.

To complete the argument, we let E be the set of all arcs of the form xz, where x is in V and z is in W.

Clearly E is a cut. Moreover, each arc xz of E is saturated and each arc zx carries a zero flow, since otherwise z would also be an element of V. If follows that the capacity of E must equal the value of ϕ, and that E is the required minimum cut.

Remark. When applying this theorem, it is often simplest to find a flow and a cut such that the value of the flow equals the capacity of the cut. It follows from the theorem that the flow must be a maximum flow and that the cut must be a minimum cut. If all the capacities are integers, then the value of a maximum flow is also an integer, this turns out to be useful in certain applications of network flows.

Problem 4.30. *Find a maximum flow in the directed network shown in Fig. (4.74) and prove that it is a maximum.*

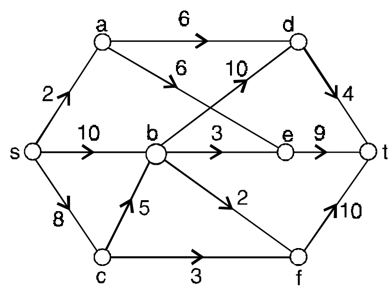


Fig. 4.74. A directed network.

Solution. We start by sending a flow of 2 units through the path *sadt*, a flow of 3 units through *sbet*, and a flow of 3 units through *scft*, obtaining the flow shown on the left in Fig. (4.75).

Continue by sending flows of 2 units through *sbdt* and 2 units through *sbft*, obtaining the flow shown on the right in Fig. (4.75)

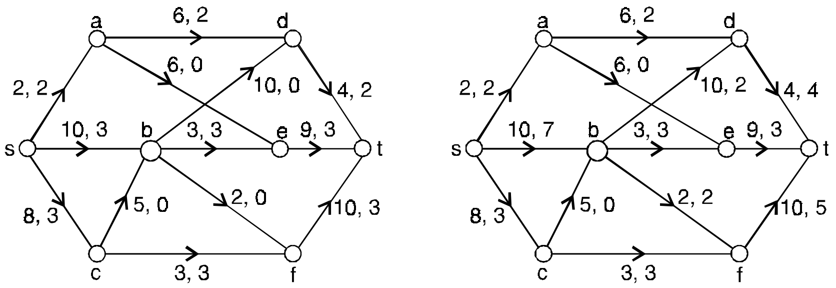


Fig. 4.75.

At this point, there are no further flow-augmenting chains from *s* to *t* involving only forward arcs.

However, we can use the backward arc *da* to obtain a flow-augmenting chain *scbdaet*.

Since the slack of this chain is 2, we add a flow of 2 to *sc*, *cb*, *bd*, *ae*, and *et*, and subtract 2 from *ad*.

The result is shown in Fig. (4.76).

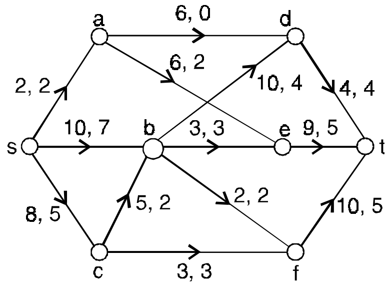


Fig. 4.76.

A search for further flow-augmenting chains takes us from *s* to *c* or *b* and on to *d*, where we are stuck.

This tells us that the current flow (of value 14) is maximum.

It also presents us with a cut verifying maximality, namely, S = {*s*, *b*, *c*, *d*} (those vertices reachable from *s* by flow-augmenting chains) and T = {*a*, *e*, *f*, *t*} (the complement of *s*).

The capacity of this cut is

$$C_{sa} + C_{be} + C_{bf} + C_{cf} + C_{dt} = 2 + 3 + 2 + 3 + 4 = 14.$$

Since this is the same as the value of the flow, we have verified that our flow is maximum.

Problem 4.31. *Why does the procedure just described of adding an amount q to the forward arcs of a chain and subtracting the same amount from the backward arcs preserve conservation of flow at each vertex ?*

Solution. The flow on the arcs incident with a vertex not on the chain are not changed, so conservation of flow continues to hold at such a vertex. What is the situation at a vertex on the chain. Remember that a chain in a directed network is just a trail whose edges can be followed in either direction, this, each vertex on a chain is incident with exactly two arcs.

Suppose a chain contains the arcs *uv, vw* (in that order) and that the flows on these arcs before changes are f_{uv} and f_{vw}.There are essentially two cases to consider.

Case 1.

Suppose the situation at vertex v in the network is $u \to v \to w$.

In this case, both uv and vw are forward arcs, so each has the flow increased by q.

The total flow into v increases by q, but so does the total flow out of v, so there is still conservation of flow at v. (the analysis is similar if the situation at v is $u \leftarrow v \leftarrow w$).

Case 2.

The situation at v is $u \to v \leftarrow w$.

Here the flow on the forward arc uv is increased by q and the flow in the backward arc wv is decreased by q. There is no change in the flow out of v.

Neither is there any change in the flow in v since the only terms in the sum $\sum_r f_{rv}$ which change occur with $r = u$ and $r = w$, and these become, respectively, $f_{uv} + q$ and $f_{vw} - q$. (the analysis is similar if the situation at v is $u \leftarrow v \to w$).

Problem 4.32. *Verify the law of conservation at vertices a, b and d.*

Solution. The law of conservation holds at a because

$$\sum_v f_{va} = f_{sa} = 10 \text{ and } \sum_v f_{av} = f_{ac} + f_{ad} = 10 + 0 = 10$$

It holds at b because $\sum_v f_{vb} = f_{sb} = 2$ and $\sum_v f_{bv} = f_{bc} + f_{bd} = 1 + 1 = 2$

It holds at d because $\sum_v f_{vd} = f_{ad} + f_{bd} = 0 + 1 = 1$ and $\sum_v f_{dv} = f_{dt} = 1$.

Problem 4.33. *What does it mean to say that {S, T} is a partition of V ?*

Solution. To say that sets S and T comprise a partition of V is to say that S and T are disjoint subsets of V whose union is V.

Problem 4.34. *With reference to the directed network of Fig. (4.77), find a flow whose value exceeds 12.*

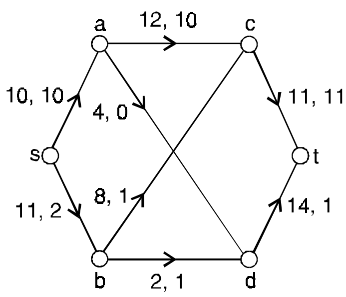


Fig. 4.77.

Solution. A flow with value 13 appears in Fig. (4.78) and one with value 17 is shown in Fig. (4.79).

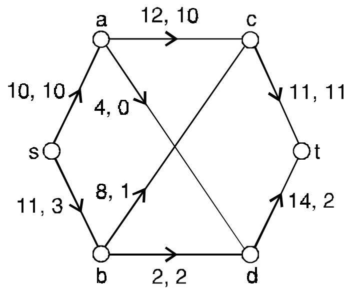


Fig. 4.78.

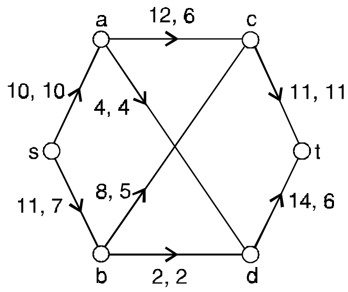


Fig. 4.79.

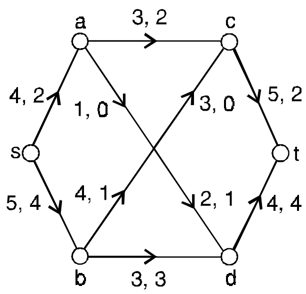
Problem 4.35. *(i) Verify the law of conservation of flow at a, e, and d.*

(ii) Find the value of the indicated flow.

(iii) Find the capacity of the (s, t)-cut defined by S = {s, a, b} and T = {c, d, e, t}

(iv) Can the flow be increased along the path sbedt ? If so, by how much ?

(v) Is the given flow maximum ? Explain.



Solution. (*i*) The law of conservation holds at a because

$$\sum_v f_{va} = f_{sa} = 2 \text{ and } \sum_v f_{av} = f_{ac} + f_{ae} = 2 + 0 = 2.$$

It holds at *e* because $\sum_v f_{ve} = f_{ae} + f_{be} = 0 + 1 = 1$

and $\sum_v f_{ev} = f_{ec} + f_{ed} = 0 + 1 = 1$

It holds at *d* because $\sum_v f_{vd} = f_{bd} + f_{ed} = 3 + 1 = 4$

and $\sum_v f_{dv} = f_{dt} = 4.$

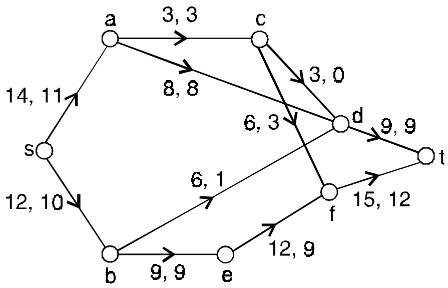
(*ii*) the value of the flow is 6.

(*iii*) The capacity of the cut is $C_{ac} + C_{ae} + C_{be} + C_{bd} = 3 + 1 + 4 + 3 = 11$.

(*iv*) No. Arc *dt* is saturated.

(*v*) The flow is not maximum. For instance, it can be increased by adding 1 to the flow in the arcs along *sact*.

Problem 4.36. *Find the capacity of the (s, t)-cut defined by S = {s, a, b, d} and T = {c, e, f, t}.*



Solution. The capacity of the cut is $C_{ac} + C_{be} + C_{dt} = 3 + 9 + 9 = 21$.

Problem 4.37. *Answer the following questions for each of the networks shown in Fig. (4.80).*

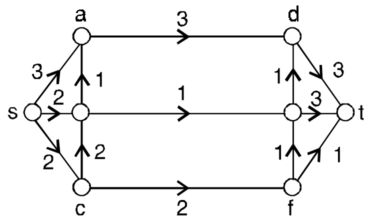


Fig. 4.80.

(i) Exhibit a unit flow

(ii) Exhibit a flow with a saturated arc.

(iii) Find a "good" and, if possible, a maximum flow in the network. State the value of your flow.

Solution. (*i*) Send one unit through the path *sbet*.

(*ii*) The flow in Fig. (4.80) has a saturated arc, *be*.

(*iii*) Here is a maximum flow, of value 6.

To see that the flow is maximum, consider the cut S = {*s*, *a*, *b*}, T = {*c*, *d*, *e*, *f*, *t*}.

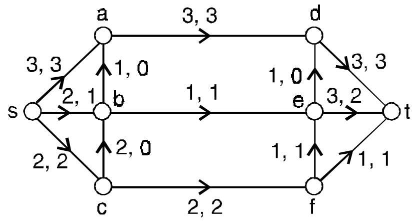
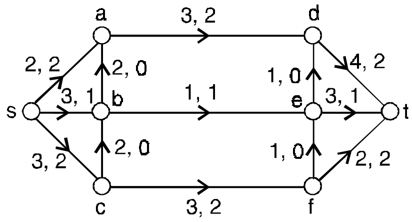


Fig. 4.81.

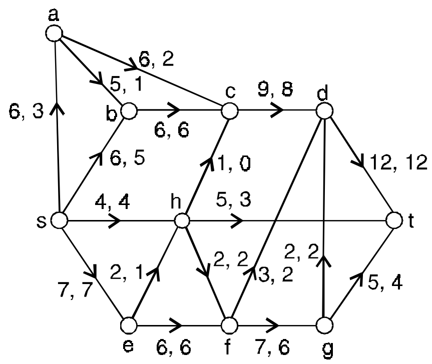
Problem 4.38. *Answer the following two questions for each of the directed networks shown.*

(i) Show that the given flow is not maximum by finding flow augmenting chain from s to t. What is the slack in your chain ?

(ii) Find a maximum flow, give its value, and prove that it is maximum by appealing to max-flow-mincut theorem.



(*a*)



(*b*)

Fig. 4.82.

Solution. (*a*) (*i*) One flow-augmenting chain is *sbadt* in which the slack is 1.

(*ii*) Here is a maximum flow, of value 7. We can see this is maximum by examining the cut S = {*s*, *a*, *b*} T = {*c*, *d*, *e*, *f*, *t*}, of capacity

$$C_{sc} + C_{ad} + C_{be} = 3 + 3 + 1 = 7, \text{ the value of the flow.}$$

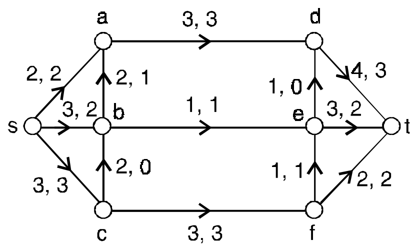


Fig. 4.83.

(*b*) (*i*) One flow-augmenting chain is *sacdfgt*, which has slack 1.

(*ii*) Here is a maximum flow, of value 20. We can see this is maximum by examining the cut S = {*s*, *a*, *b*, *c*}, T = {*d*, *e*, *f*, *g*, *h*, *t*}.

This has capacity $C_{se} + C_{sh} + C_{cd} = 7 + 4 + 9 = 20$, the value of the flow.

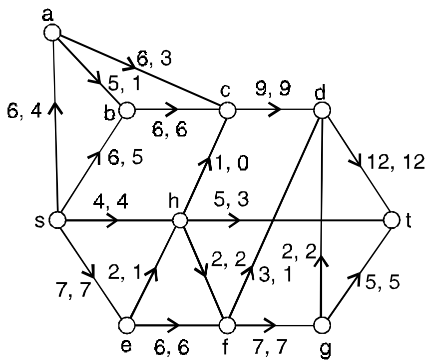


Fig. 4.84.

4.6 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 $\delta(A) = | A | - | R(A) |$ is called the deficiency of A. The deficiency of graph G, denoted $\delta(G)$, is given by $\delta(G) = \max \{\delta(A)/A \subseteq X\}$.

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.

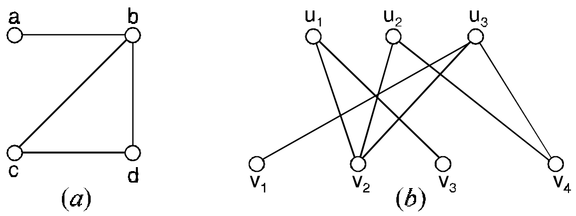


Fig. 4.85.

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 $\{b, d\}$: $a \rightarrow b$, $c \rightarrow d$.

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.

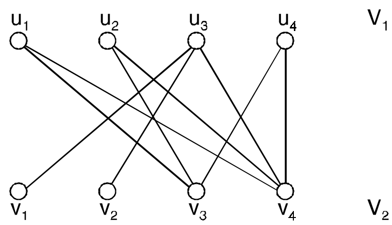


Fig. 4.86.

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.

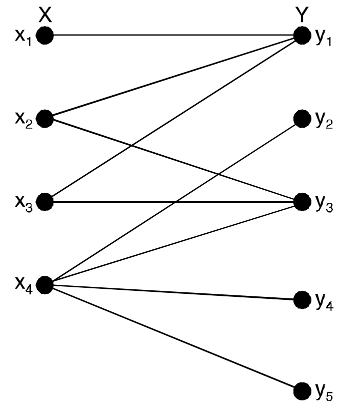


Fig. 4.87.

Theorem 4.18. *Let G = (V, E) be bipartite with V partitioned as X $\cup$ Y. A complete matching of X into Y exists if and only if for every subset of X, | A | $\le$ | R(A) |, where R(A) is the subset of Y consisting of those vertices each of which is adjacent to at least one vertex in A.*

Proof. With V partitioned as X $\cup$ Y, let X = $\{x_1, x_2, ..., x_m\}$ and Y = $\{y_1, y_2, ..., y_n\}$

Construct a transport network N that extends graph G by introducing two new vertices a and z (the source and sink).

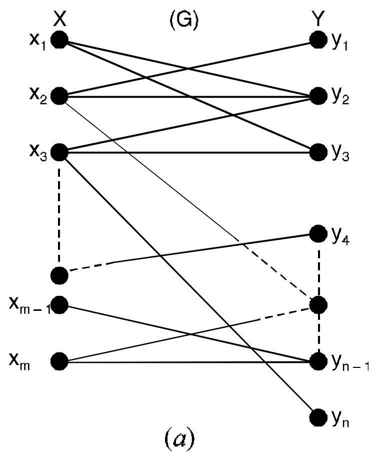
For each vertex x_i, $1 \le i \le m$, draw edge (a, x_i) ; for each vertex y_j, $1 \le j \le n$, draw edge (y_j, z).

Each new edge is given a capacity of 1. Let M be any positive integer that exceeds | X |. Assign each edge in G the capacity M.

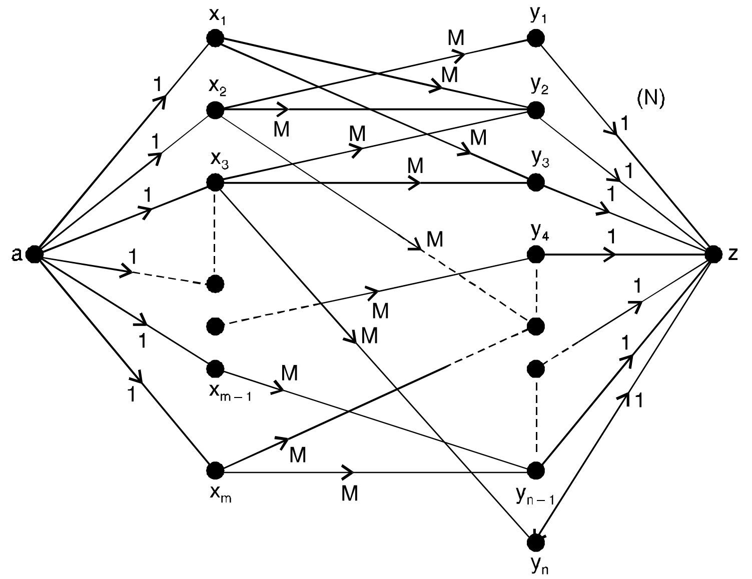
The original graph G and its associated network N appear as shown in Fig. (4.88).

It follows that a complete matching exists in G if and only if there is a maximum flow in N that uses all edges (a, x_i), $1 \le i \le m$.

Then the value of such a maximum flow is m = | X |.



(*a*)



(*b*)

Fig. 4.88.

We shall prove that there is a complete matching in G by showing that C(P, $\bar{P}$) ≥ | X | for each cut (P, $\bar{P}$) in N. So if (P, $\bar{P}$) is an arbitrary cut in the transport network N, let us define A = X ∩ P and B = Y ∩ P.

Then A ⊆ X where we shall write A = {$x_1, x_2, ..., x_i$} for some $0 \le i \le m$.

Now P consists of the source a together with the vertices in A and the set B ⊆ Y, as shown in Fig. (4.89)(*a*).

In addition, $\bar{P}$ = (X – A) ∪ (Y – B) ∪ {*z*}.

Since each of these edges has capacity 1, C(P, $\bar{P}$) = | X – A | + | B | = | X | – | A | + | B |, with B ⊇ R(A), we have | B | ≥ R(A), and since | R(A) | ≥ | A |, it follows that | B | ≥ | A |.

Consequently, *c*(P, $\bar{P}$) = | X | + (| B | – | A |) ≥ | X |.

Therefore, since every cut in network N has capacity at least | X |, such a flow will result in exactly | X | edges from X to Y having flow 1, and this flow provides a complete matching of X into Y.

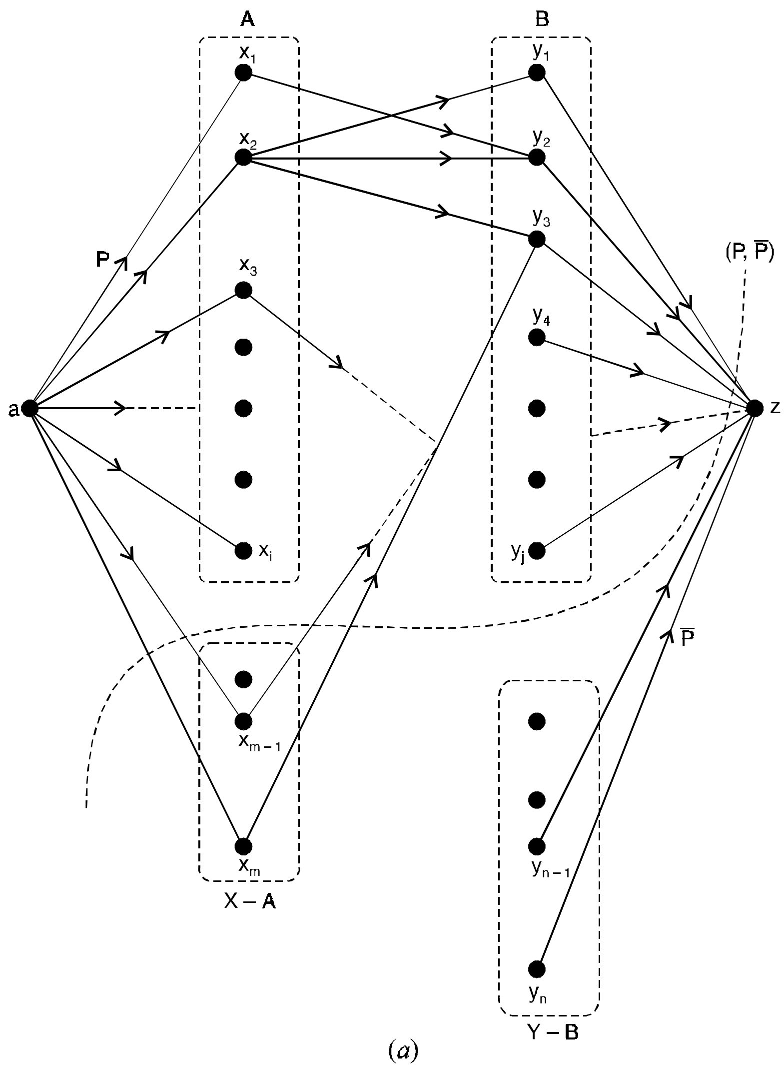
Conversely, suppose that there exists a subset A of X where | A | > | R(A) |.

Let (P, $\bar{P}$) be the cut shown for the network in Fig. (4.89)(*b*), with P = {*a*} ∪ A ∪ R(A) and $\bar{P}$ = (X – A) ∪ (Y – R(A)) ∪ {*z*}. Then C(P, $\bar{P}$) is determined by (*i*) the edges from the source *a* to the vertices in X – A and (*ii*) the edges from the vertices in R(A) to the sink *z*.

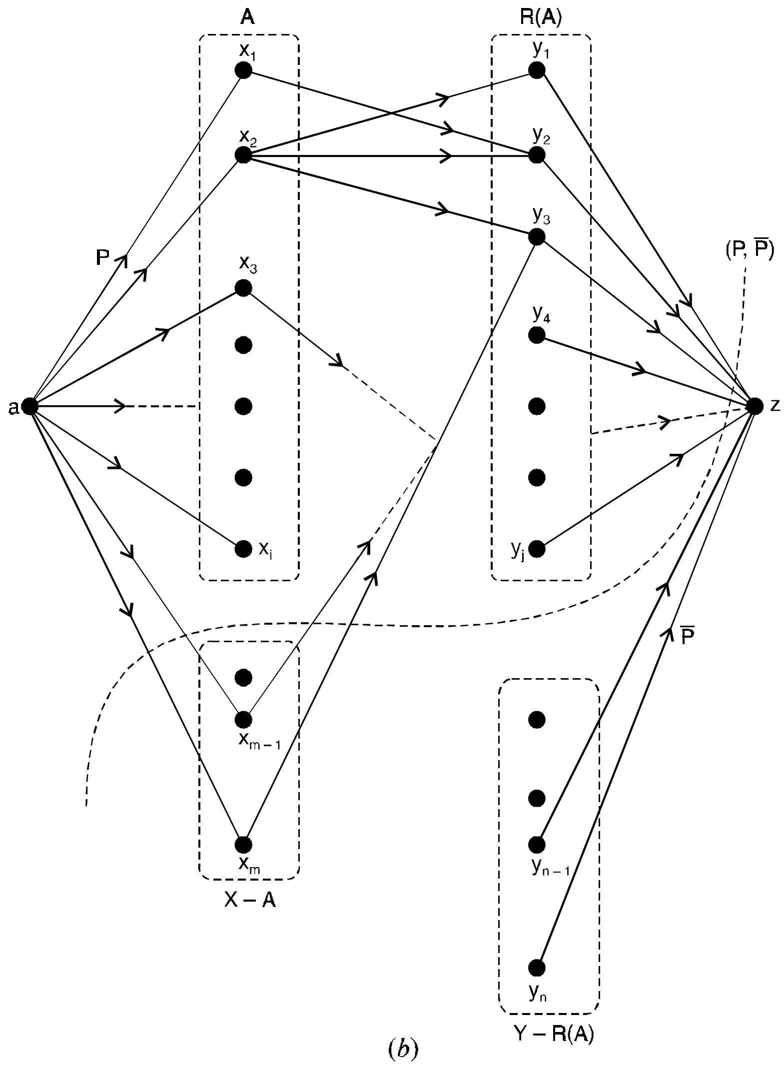
Hence C(P, $\overline{\text{P}}$) = | X – A | + | R(A) | = | X | – (| A | – | R(A) |) < | X |,

since | A | > | R(A) |. The network has a cut of capacity less than | X |, it follows that any maximum flow in the network has value smaller than | X |.

Therefore, there is no complete matching from X into Y for the given bipartite graph G.



(*a*)



(*b*)

Fig. 4.89

Theorem 4.19. *For any bipartite graph G with partition V_1 and V_2, if there exists a positive integer m satisfying the condition that $deg_G(v_1) \geq m \geq deg_G(v_2)$, for all vertices $v_1 \in V_1$ and $v_2 \in V_2$, then a complete matching of V_1 into V_2 exists.*

Proof. Let G be a bipartite graph with partition V_1 and V_2.

Let m be a positive integer satisfying the condition that $\deg_G(v_1) \geq m \geq \deg_G(v_2)$ for all vertices $v_1 \in V_1$ and $v_2 \in V_2$.

Consider an r-element subset S of the set V_1.

Since the $\deg(v_1) \geq m$, from each element of S, there are at least m edges incident to the vertices in V_1.

Thus there are $m\ r$ edges incident from the set S to the vertices in V_1, but degree of every vertex of V_2 cannot exceed m implies that these $m\ r$ edges are incident on at least $(mr)/r = r$ vertices in V_2.

Hence, there exists a complete matching of V_1 into V_2 exists.

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 $|X| \leq |A(X)|$.

Problem 4.40. *Can you conclude from this problem that G also has a matching which saturates V_2 ? More generally, does G have a matching which saturates both V_1 and V_2 at the same time (a perfect matching) ?*

Solution. Yes, the same argument works. But more easily, note that since G is bipartite, the sum of the degrees of vertices in V_1 must equal the sum of degrees of vertices in V_2.

Since all vertices have the some degree, we conclude that $| V_1 | = | V_2 |$, so a matching which saturates V_1 must automatically saturate V_2 as well and vice versa.

Proposition : 4.1. Let G be a graph with vertex set V.

1. If G has a perfect matching then $| V |$ is even.
2. If G has a Hamiltonian path or cycle then G has a perfect matching if and only if $| V |$ is even.

Theorem 4.21. *If G is a graph with vertex set V, $| V |$ is even, and each vertex has degree $d \geq \frac{1}{2} | V |$ then G has a perfect matching.*

Problem 4.41. *Given a set S and n subsets $A_1, A_2, \ldots, A_n$ of S, it is possible to select distinct elements $s_1, s_2, \ldots, s_n$ of S such that $S_1 \in A_1, S_2 \in A_2, \ldots, S_n \in A_n$ if and only if, for each subset X of $\{1, 2, \ldots, n\}$ the number of elements in $\bigcup_{x \in X} A_x$ is at least $| X |$. Why ?*

Solution. Construct a bipartite graph with vertex sets V_1 and V_2 where V_1, has n vertices corresponding to $A_1, A_2, \ldots, A_n$, V_2 has one vertex for each element of S and there is an edge joining A_i to s if and only if $s \in A_i$.

Given a subset X of V_1, the set A(X) is precisely the set of elements in $\bigcup_{x \in X} A_x$.

Thus this question is just a restatement of Hall's Marriage theorem.

Problem 4.42. *Determine necessary and sufficient conditions for the complete bipartite graph $K_{m, n}$ to have a perfect matching.*

Solution. $K_{m, n}$ has a perfect matching if and only if $m = n$. To see this, first assume that $m = n$ and let the vertex sets be $V_1 = \{u_1, u_2, \ldots, u_m\}$ and $V_2 = \{v_1, v_2, \ldots, v_m\}$.

Then $\{u_1v_1, u_2v_2, \ldots, u_mv_m\}$ is a perfect matching.

Conversely, say we have a perfect matching and $m \leq n$. Since each edge in a matching must join a vertex of V_1 to a vertex of V_2, there can be at most m edges.

If $m < n$, some vertex in V_2 would not be part of any edge in the matching, a contradiction.

Thus, $m = n$.

Problem 4.43. *Show that a complete matching of V_1 into V_2 exists in the following graph.*

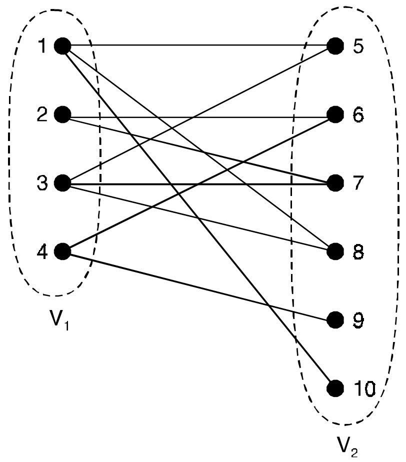


Fig. 4.90.

Solution. The minimum degree of a vertex of $V_1 = 2 \geq 2$

= Maximum degree of a vertex of V_2

By choosing $m = 2$, there exists a complete matching from the set V_1 into V_2.

Problem 4.44. *Find whether a complete matching of V_1 into V_2 exist for the following graph ? What can you say from V_2 into V_1.*

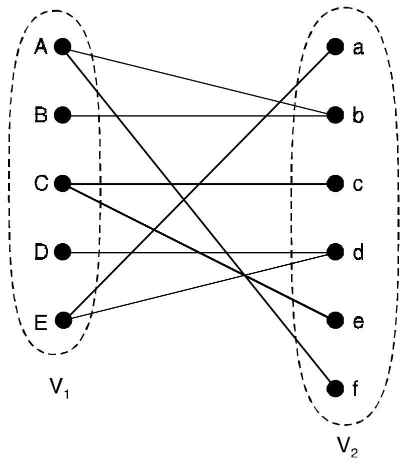


Fig. 4.91.

Solution. Yes, a complete matching exists from V_1 into V_2, which is {A*f*, B*b*, C*c*, D*d*, E*a*}.

This matching is not unique, because {A*f*, B*b*, C*e*, D*d*, E*a*} is also a complete matching from V_1 into V_2 complete matching from V_2 into V_1 is not exists because cordinality of V_2 is more than the cordinlity of V_1.

Problem 4.45. *Find whether a complete matching of V_1 into V_2 exist for the following graph.*

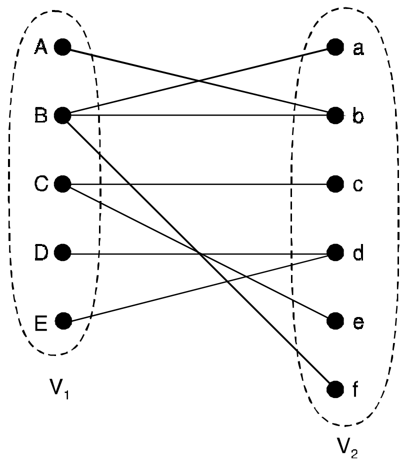


Fig. 4.92.

Solution. No, because if we take a subset {D, E} of V_1 having two vertices, then the elements of this set is collectively adjacent to only the subset $\{d\}$ of V_2.

The cordinality of $\{d\}$ is one that is less than the cordinality of the set {D, E}.

Problem 4.46. *Find a complete matching of the graph of Fig. (4.93).*

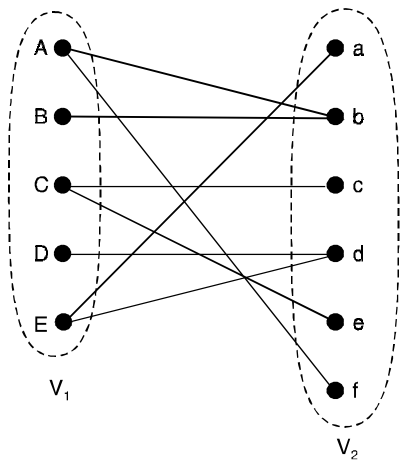


Fig. 4.93.

Solution. $X(G) = \begin{bmatrix} 0 & X_{n_1 \times n_2} \\ X^T_{n_1 \times n_2} & 0 \end{bmatrix}$ where $X_{n_1 \times n_2} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$

Here $n_1 = 5$, $n_2 = 6$ and $n = n_1 + n_2 = 11$ = total number of vertices of G.

Step 1 : Choose the row B and the column *b* (since B contains 1 in only one place in the entire row).

Step 2 : Discard the column *b* (since it is already chosen).

Step 3 : Choose the row D and the column *d* (since D contains 1 in only one place in the entire row).

Step 4 : Discard the column *d* (since it is already chosen).

Step 5 : Choose the row E and the column *a* (since E *a*th entry is one and which are not chosen earlier).

Step 6 : Discard the column *a* (the edge which is chosen in step 5).

Step 7 : Choose the row A and the column *f* (since the row A contains exactly one 1 in the column *f*).

Step 8 : Discard the column *f* (since it is chosen in step 7).

Step 9 : Choose the row C and the column the column *e* (or *c*) (since C is the final).

Step 10 : No row is left to choose and all the rows are able to choose, hence the matching is complete.

The resultant matrices after each step and the final matching is given below.

After the steps 1 and 2

$$\begin{array}{c} \\ A \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} a & c & d & e & f \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

After the steps 3 and 4

$$\begin{array}{c} \\ A \\ C \\ E \end{array}\begin{array}{c} \begin{array}{cccc} a & c & e & f \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

After the steps 5 and 6

$$\begin{array}{c} \\ A \\ C \end{array}\begin{array}{c} \begin{array}{ccc} c & e & f \end{array} \\ \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \end{array}$$

After the steps 7 and 8

$$C \begin{bmatrix} c & e \\ 1 & 1 \end{bmatrix}$$

Resultant matrix and the corresponding matching are shown in Fig. (4.94).

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{cccccc} a & b & c & d & e & f \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

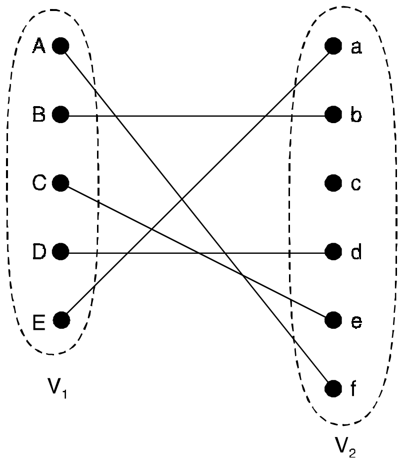


Fig. 4.94.

The complete matching is {Af, Bb, Ce, Dd, Ea}.

Problem 4.47. *Prove that the bipartite graph shown in Fig. 4.95. does not have a complete matching.*

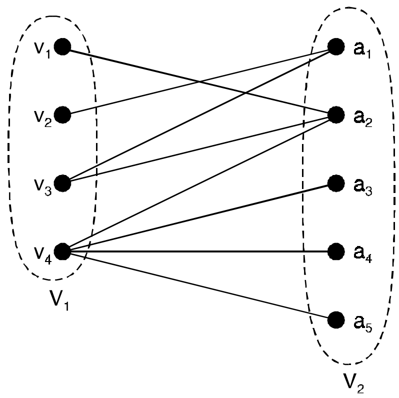


Fig. 4.95.

Solution. We observe that the three vertices v_1, v_2, v_3 in V_1 are together joined to two vertices a_1, a_2, in V_2. Thus, there is a subset of 3 vertices in V_1 which is collectively adjacent to 2 (< 3) vertices in V_2.

Hence, by Hall's theorem, there does not exist a complete matching from V_1 to V_2.

Problem 4.48. *Show that for the graph in Fig. (4.95) there does not exist a positive integer m such that the degree of every vertex in $V_1 \geq m \geq$ the degree of every vertex in V_2.*

Solution. From the graph, we find that degree of $v_1 = 1$ and degree of $a_2 = 3$

Therefore, the specified condition does not hold for any positive integer m.

That this is indeed the situation is confirmed by the fact that in this graph there is no complete matching from V_1 to V_2.

Problem 4.49. *Three boys b_1, b_2, b_3 and four girls g_1, g_2, g_3, g_4 are such that (i) b_1 is a cousin of g_1, g_3, g_4. (ii) b_2 is a cousin of g_2 and g_4 (iii) b_3 is a cousin of g_2 and g_3.*

Can every one of the boys marry a girl who is one of his cousins ? If so, find possible sets of such couples.

Solution. Let us draw a bipartite graph $G(V_1, V_2 ; E)$ in which V_1 consists of b_1, b_2, b_3 and V_2 consists of g_1, g_2, g_3, g_4 and E consists of edges representing the cousin relationship. The graph is as shown in Fig. (4.96).

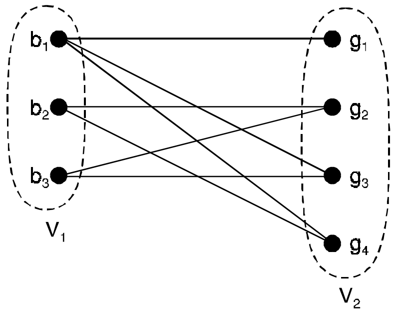


Fig. 4.96.

The problem is one of finding whether a complete matching exists from V_1 and V_2.

We have to consider every subset of V_1 with $k = 1, 2, 3$ elements and find whether each subset is collectively adjacent to k or more vertices in V_2. The subsets S_i of V_1 and their collective adjacent subsets S_i' in V_2 are shown in the following table :

K	S_i	S_i'
$k = 1$	$\{b_1\}$	$\{g_1, g_3, g_4\}$
	$\{b_2\}$	$\{g_2, g_4\}$
	$\{b_3\}$	$\{g_2, g_3\}$
$k = 2$	$\{b_1, b_2\}$	$\{g_1, g_2, g_3, g_4\}$
	$\{b_1, b_3\}$	$\{g_1, g_2, g_3, g_4\}$
	$\{b_2, b_3\}$	$\{g_2, g_3, g_4\}$
$k = 3$	$\{b_1, b_2, b_3\}$	$\{g_1, g_2, g_3, g_4\}$

We observe that, for each S_i, the number of elements in S_i' is greater than or equal to the number of elements in S_i.

Therefore, the graph has a complete matching. This means that each boy can marry a girl who is one of this cousins.

By exammining the graph in Fig. (4.96) or the table above, we find the following five possible couple sets :

Step 1 : $(b_1, g_1), (b_2, g_2), (b_3, g_3)$

Step 2 : $(b_1, g_1), (b_2, g_4), (b_3, g_2)$

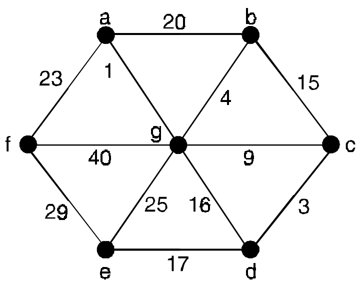
Step 3 : $(b_1, g_1), (b_2, g_4), (b_3, g_3)$

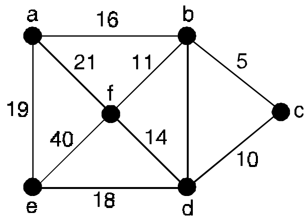
Step 4 : $(b_1, g_1), (b_2, g_4), (b_3, g_2)$

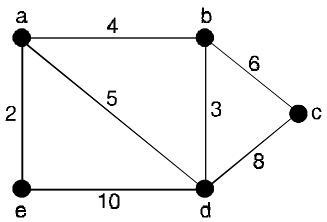
Step 5 : $(b_1, g_4), (b_2, g_2), (b_3, g_3)$.

Problem Set 4.1

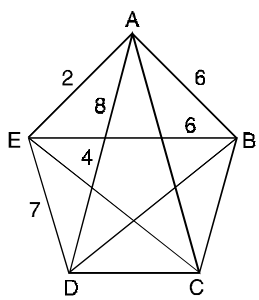
1. Use Krushal's algorithm to find a minimum spanning tree for the given weighted graphs :

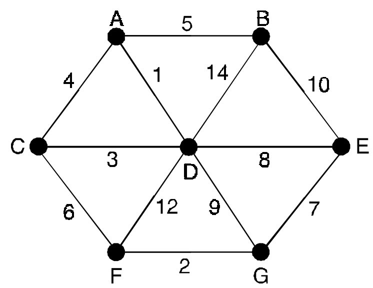
(*a*) 

(*b*) 

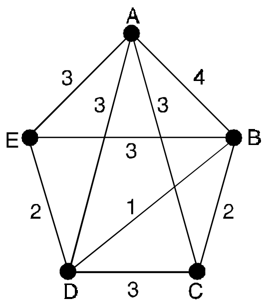
(*c*) 

2. Use Prim's algorithm to find a minimum spanning tree for the given weighted graphs

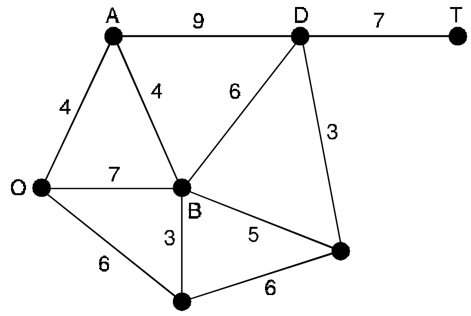
(*a*) 

(*b*) 

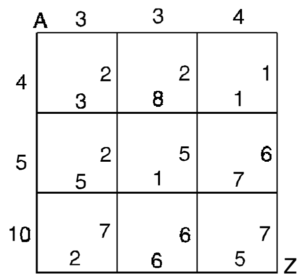
(*c*)



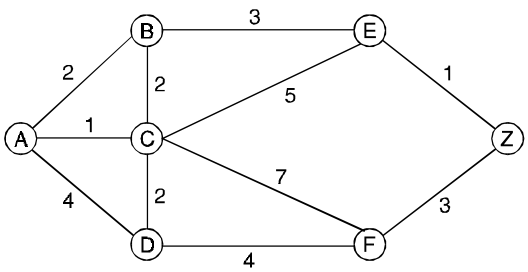
(*d*)



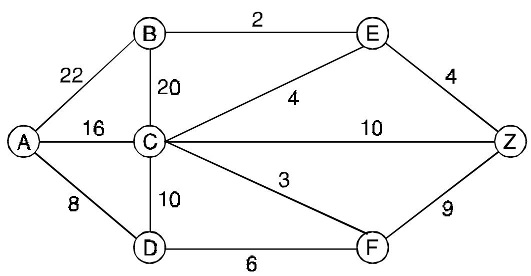
3. Using Dijsktra's algorithm find the shortest path between node A and node Z in the mesh graph of the figure :



4. Apply Dijsktra's algorithm to find shortest path between node A and node Z in the graph shown in the Figure (4.97) (*a*) and (*b*) below



(*a*)



(*b*)

5. Apply Dijsktra's algorithm to find shortest path from node A to all other nodes in the graph show in the figure (*a*) and (*b*) above.

6. Apply Floyd Warshall's algorithm to compute all pair shortest distance in the graphs shown in the figure (4.97) (*a*) and (*b*) above.

7. Find spanning tree using Prim's algorithm for the graph of figures (4.98), (4.99) and (4.100).

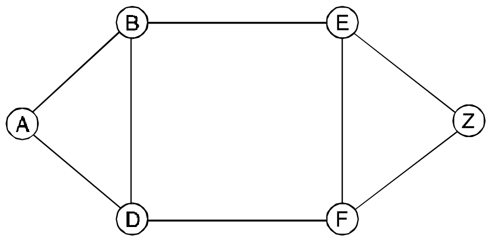


Fig. 4.98.

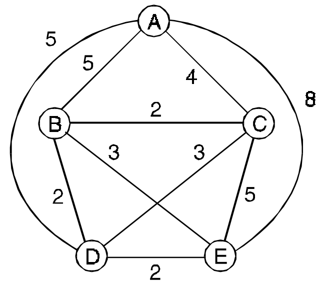


Fig. 4.99.

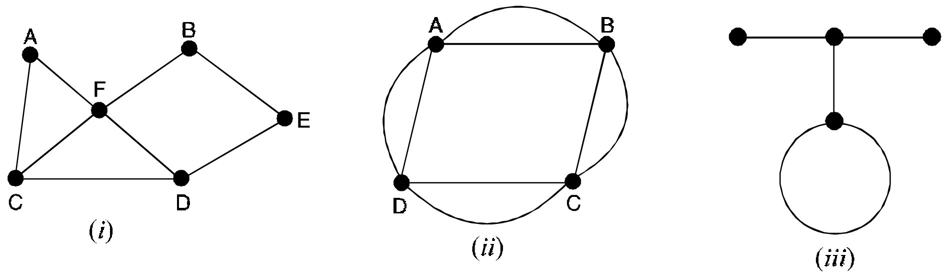


Fig. 4.100.

8. Using Prim's algorithm, find minimum spanning tree from the graphs of the figure

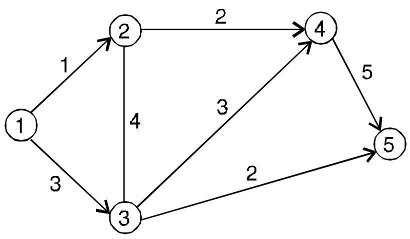


Fig. 4.101.

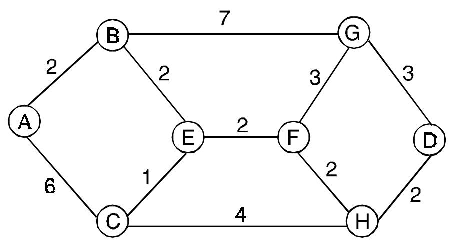


Fig. 4.102.

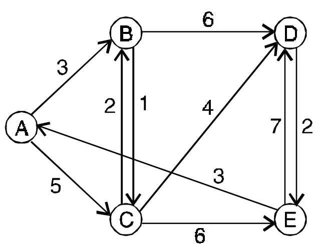


Fig. 4.103.

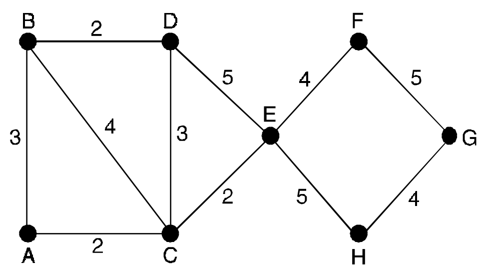
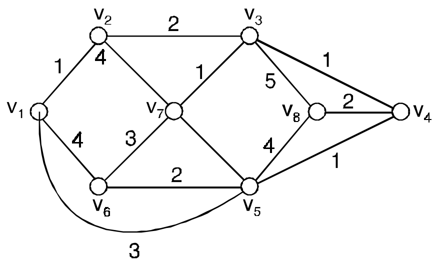


Fig. 4.104.

9. Use Kruskal's algorithm to find minimum spanning tree from the graphs of the figures (4.101), (4.102), (4.103) and (4.104).

10. The Floyd-Warshall algorithm is applied to the graph shown.

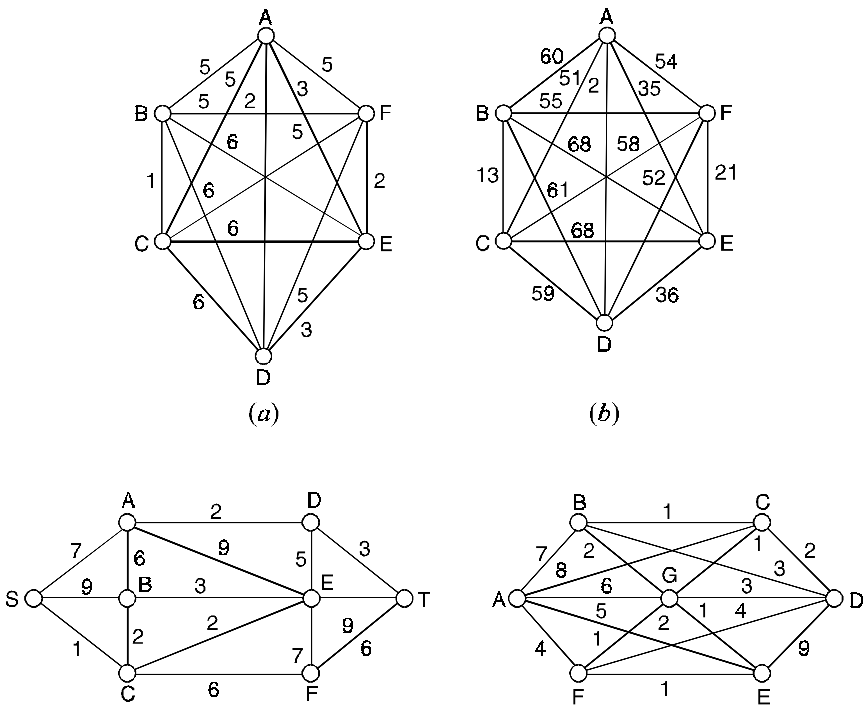


(*a*) Find the final values of $d(7, 1)$, $d(7, 2)$, ... $d(7, 8)$.

(*b*) Find the values of $d(1, 2)$, $d(3, 4)$, $d(2, 5)$ and $d(8, 6)$ after $k = 4$.

(*c*) Find the values of $d(6, 8)$ at the start and as k varies from 1 to 8.

11. Use Kruskal's algorithm to find a spanning tree of minimum total weight in each of the graphs in Figure (4.105). Give the weight of your minimum tree and show your steps.



(*a*) (*b*)

(*c*) (*d*)

12. Suppose we have a connected graph G and we want to find a spanning tree for G which contains a given edge *e*. How could Kruskal's algorithm to do this ? Discuss both the weighted and unweighted cases.

13. Prove that at each vertex v of a weighted connected graph, Kruskal's algorithm always includes an edge of lowest weight incident with v.

14. Suppose v_1, v_2 are the bipartition sets of a bipartition graph G. Let m be the smallest of the degrees of the vertices in v_1 and M the largest of the degrees of the vertices in v_2. Prove that if $m \geq$ M, then G has a matching which saturates v_1.

15. Suppose v_1 and v_2 are the bipartition sets in a bipartite graph G. If $| v_1 | > | v_2 |$, then it is clearly impossible to find a matching which saturates v_1. State a result which is applicable to this case and give a necessary and sufficient condition for your result to hold.

16. Show that the complete graph K_n has a perfect matching if and only if n is even.

17. Prove that if a tree has a perfect matching, then that matching is unique.

18. Find a maximal (s, t)-flow. Verify your answer by finding an (s, t)-cut whose capacity equals the value of the flow.

19. Let G be a connected graph, every vertex of which has degree 2. Prove that G has a perfect matching if and only if G has an even number of vertices.

20. Apply Dijkstra's algorithm to the weighted directed multipgraph shown in Figure below, and find the shortest distance from vertex a to the other seven vertices in the graph.

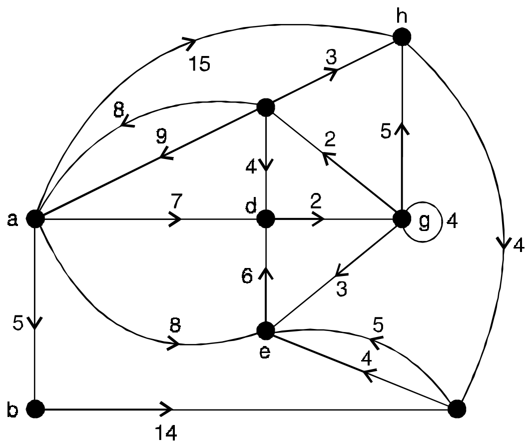


Fig. 4.106.

21. For $n \in z^+$ and for each $1 \leq i \leq n$, let $A_i = \{1, 2, 3, ..., n\} - \{i\}$. How many different systems of distinct representative exist for the collection $A_1, A_2, A_3, ... A_n$?

22. Using the concept of flow in a transport network, construct a directed multigraph G = (V, E), with V = $\{u, v, w, x, y\}$ and $id(u) = 1$, $od(u)= 3$; $id(v) = 3$, $od(v) = 3$; $id(w) = 3$, $od(w) = 4$; $id(x) = 5$, $od(x) = 4$; and $id(y) = 4$, $od(y) = 2$.

23. (*a*) Determine all systems of distinct representatives for the collection of sets

$$A_1 = \{1, 2\}, A_2 = \{2, 3\}, A_3 = \{3, 4\}, A_4 = \{4, 1\}.$$

(*b*) Given the collection of sets $A_1 = \{1, 2\}, A_2 = \{2, 3\}, ..., A_n = \{n, 1\}$, determine how many different systems of distinct representatives exist for the collection.

24. For the graph shown in Figure below, if four edges are selected at random, what is the probability that they provide a complete matching of X into Y ?

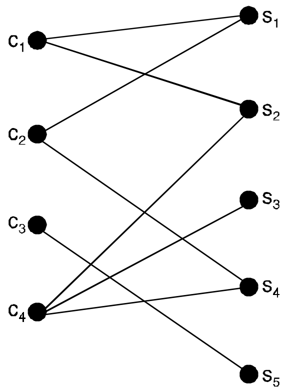


Fig. 4.107.

25. Let G = (V, E) be a bipartite graph where V is partitioned as $X \cup Y$. If $\deg(x) \geq 4$ for all $x \in X$ and $\deg(y) \leq 5$ for all $y \in Y$, prove that if $|X| \leq 10$ then $\delta(G) \leq 2$.

26. Find a maximum flow and the corresponding minimum cut for each transport network shown in Figure below.

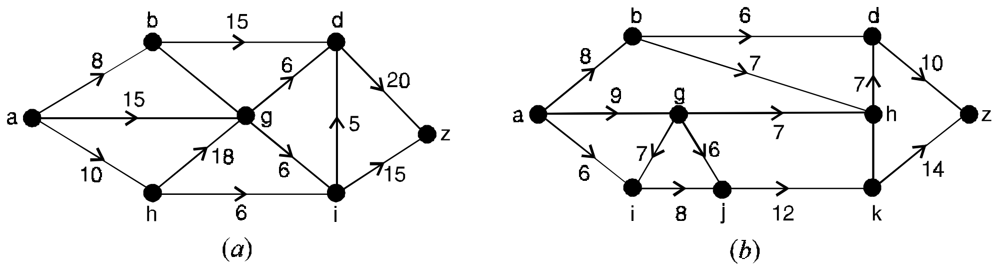


Fig. 4.108.

27. Using Prim's algorithm, find a minimal spanning tree for the following weighted graph.

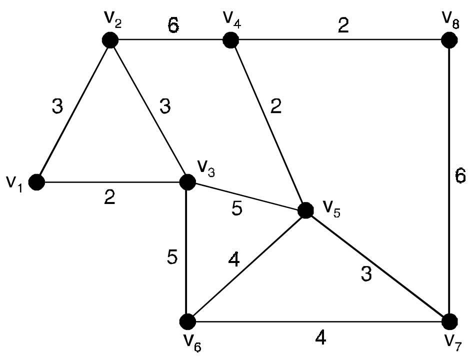


Fig. 4.109.

28. Using Prim's algorithm, find a minimal spanning tree of the weighted graph in Figure given below :

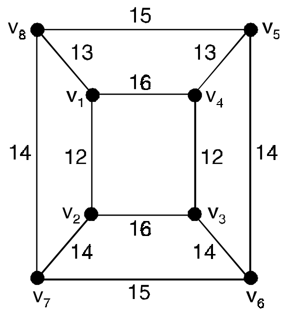


Fig. 4.110.

29. Using Prim's algorithm, find a minimal spanning tree for the weighted graph shown below :

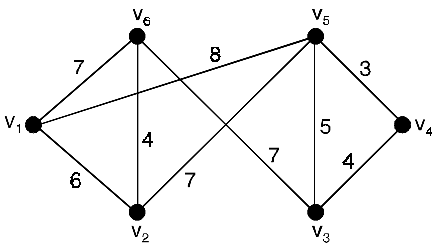


Fig. 4.111.

30. Using Prim's algorithm, find a minimal spanning tree and its weight for the weighted graph shown below :

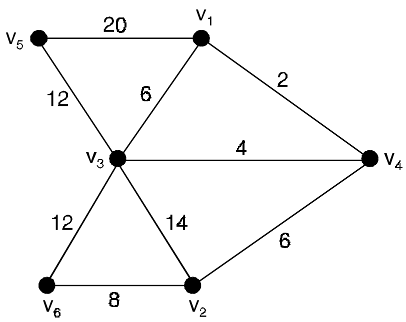


Fig. 4.112.

Answers 4.1

1. (*a*) Choose *af, ag, gb, gc, cd, de*

(*b*) Choose *ab, bf, bc, bd, de*

(*c*) Choose *ab, ae, bc, bd*

2. (*a*) Choose AE, AC, DC, AB

(*b*) Choose BA, AD, DC, CF, FG, GE

(*c*) Choose AC, CB, BD, DE

(*d*) Choose OA, AB, BC, BF, FD, DT.

11. (*a*) We want five edges (since there are six vertices).

Choose BC, then AD, FE and DE. We would like next to choose AE, but this would complete a circuit with AD and DE, so we choose AC and obtain the spanning tree shown, of weight 13.

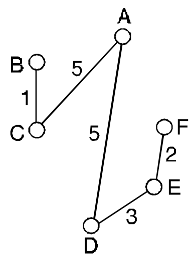


Fig. 4.113.

12. If the graph is unweighted, put a weight of 1 on edge *e* and 2 on every other edge. If the graph is weighted, ensure that the weights of the edges different from *e* are all larger than the weight of *e*. In either case, Kruskal's algorithm will select *e* first.

13. Let *d* be the lowest weight among the edges incident with vertex *v*. The first time that an edge incident with *v* is considered for selection, no edge incident with *v* will complete a circuit with edges previously selected. Always seeking edges of lowest weight, the algorithm must select an edge of weight *d*.

15. Exchange the roles of v_1 and v_2. It may be possible to find a matching which saturates v_2. Here's a necessary and sufficient condition for this to happen. If X is any subset of v_2 and A(X) is the set of all vertices of v_1 which are adjacent to some vertex of X, then $| A(X) | \geq | X |$.

16. If K_n has a perfect matching, then *n* must be even. Conversely, assume that $n = 2m$ is even and label the vertices $u_1, u_2, \ldots\ldots u_{2m}$. Since every pair of vertices is joined by an edge

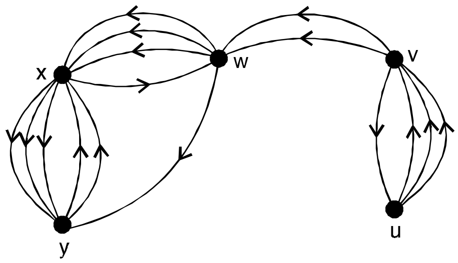
$\{u_1u_2, u_3u_4, \ldots\ldots, u_{2m-1}u_{2m}\}$ will give us a perfect matching.

20. $d(a, b) = 5$, $d(a, c) = 11$, $d(a, d) = 7$, $d(a, e) = 8$, $d(a, f) = 19$, $d(a, g) = 9$, $d(a, h) = 14$.

Note that the loop at vertex *g* and the edges (*c*, *a*) of weight 9 and (*f*, *e*) of weight 5 are of no significance.

21. There are d_n, the number of derangements of $\{1, 2, 3, \ldots.., n\}$.

22.



23. (*a*) (*i*) Select i from A_i for $1 \le i \le 4$.

(*ii*) Select $i + 1$ from A_i for $1 \le i \le 3$, and 1 from A_4.

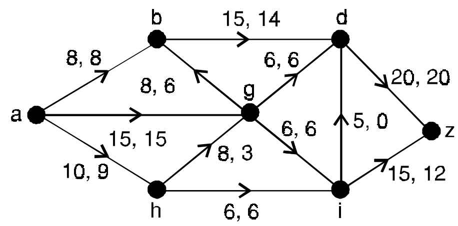
(*b*) 2.

24. $5 \Big/ \binom{8}{4} = 1/14.$

25. For each subset A of X, let G_A be the subgraph of G induced by the vertices in $A \cup R(A)$. If e is the number of edges in G_A, then $e \ge 4 \mid A \mid$ because $\deg(a) \ge 4$ for $a \in A$. Likewise, $e \le 5 \mid R(A) \mid$ because $\deg(b) \le 5$ for all $b \in R(A)$. So $5 \mid R(A) \mid \ge 4 \mid A \mid$ and $\delta(A) = \mid A \mid - \mid R(A) \mid \le \mid A \mid - (4/5) \mid A \mid = (1/5) \mid A \mid \le (1/5) \mid X \mid = 2$.

Then since $\delta(G) = \max\{\delta(A) \mid A \subseteq X\}$, we have $\delta(G) = 2$.

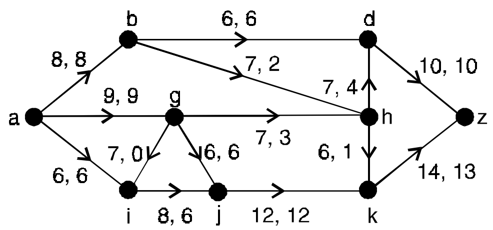
26. (*i*)



(*a*)

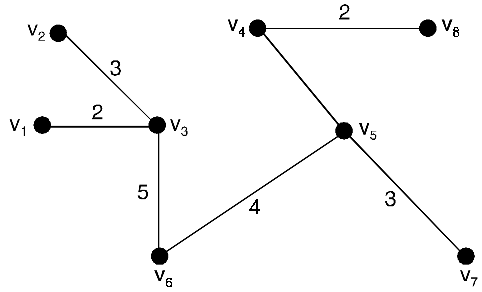
The maximum flow is 32 which is $C(P, \bar{P})$ for $P = \{a, b, d, g, h\}$ and $\bar{P} = \{i, z\}$

(*ii*)

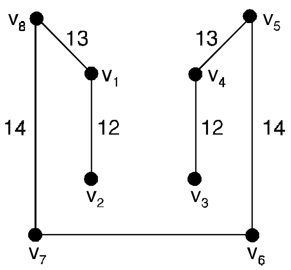


The maximum flow is 23, which is $C\{P, \bar{P}\}$ for $P = \{a\}$ and $\bar{P} = \{b, g, i, j, d, h, k, z\}$.

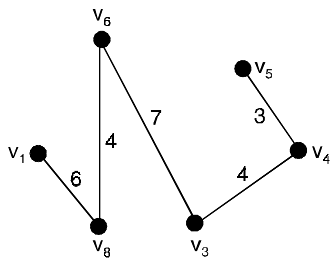
27.


v2
3
v4
2
v8
v1
2
v3
v5
5
4
3
v6
v7

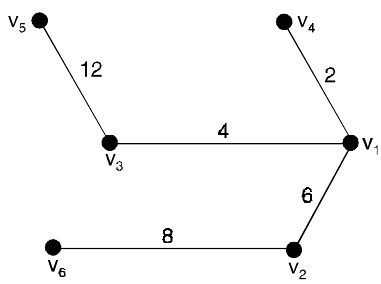
28.


v8
13
13
v5
v1
v4
14
12
12
14
v2
v3
v7
v6

29.


v6
v5
7
3
v1
4
v4
6
4
v8
v3

30.


v5
v4
12
2
4
v1
v3
6
8
v6
v2

CHAPTER

5 Matroids and Transversal Theory

INTRODUCTION

The idea of a matroid, first studied in 1935 in a pioneering paper by Hassler Whitney. A matroid is a set with an independence structure.

We defined a spanning tree in a connected graph G to be a connected subgraph of G that contains no cycles and includes every vertex of G. We note that a spanning tree cannot contain another spanning tree as a proper subgraph.

Note : (*i*) If B_1 and B_2 are spanning trees of G and e is an edge of B_1 then there is an edge f in B_2 such that $(B_1 - \{e\}) \cup \{f\}$ is also a spanning tree of G.

(*ii*) If V is a vector space and if B_1 and B_2 are bases of B and e is an element of B_1 then we can find an element f of B_2 such that $(B_1 - \{e\}) \cup \{f\}$ is also a basis of V.

5.1 MATROID

A matroids M consists of a non-empty finite set E and a non-empty collection B of subsets of E, called bases satisfying the following properties

B(*i*) no base properly contains another base

B(*ii*) if B_1 and B_2 are bases and if e is any element of B_1 then there is an element f of B_2 such that $(B_1 - \{e\}) \cup \{f\}$ is also a base.

Note : Any two bases of a matroid M have the same number of elements, this number is called the rank of M.

5.2 CYCLE MATROID

A matroid can be associated with any graph G by letting E be the set of edges of G and taking as bases the edges of the spanning forests of G, this matroid is called the cycle matroid of G and is denoted by M(G).

5.3 VECTOR MATROID

If E is a finite set of vectors in a vector space V, then we can define a matroid on E by taking as bases all linearly independent subsets of E that span the same subspace as E. A matroid obtained in this way is called a vector matroid.

5.4 INDEPENDENT (MATROIDS) SETS

A subset of E is independent if it is contained in some base of the matroid M.

(*i*) for a vector matroid, a subset of E is independent whenever its elements are linearly independent as vectors in the vector space.

(*ii*) for the cycle matroid, M(G) of a graph G, the independent sets are those sets of edges of G that contain no cycle. *i.e.*, the edge sets of the forests contained in G.

Since the bases of M are maximal independent sets, a matroid is uniquely defined by specifying its independent sets.

5.5 MATROID (MODIFIED DEFINITION)

A matroid M consists of a non-empty finite set E and a non-empty collection I of subsets of E, called independent sets, satisfying the following properties :

I(*i*) any subset of an independent set is independent

I(*ii*) if I and J are independent sets with | J | > | I | then there is an element *e*, contained in J but not in I such that $I \cup \{e\}$ is independent.

Note : (*i*) A base is defined to be a maximal independent set.

(*ii*) Any independent set can be extended to a base.

5.6 DEPENDENT (MATROID) SETS

If M = (E, I) is a matroid defined in terms of its independent sets then a subset of E is dependent if it is not independent and a minimal dependent set is called a cycle.

If M(G) is the cycle matroid of a graph G then the cycles of M(G) are precisely the cycles of G. Since a subset of E is independent if and only if it contains no cycles, a matroid can be defined in terms of its cycles.

5.7 RANK OF A

If M = (E, I) is a matroid defined in terms of its independent sets and if A is a subset of E then the rank of A denoted by $r(A)$, is the size of the largest independent set contained in A.

We note that the rank of M is equal to $r(E)$ since a subset A of E is independent if and only if $r(A) = |A|$.

Note : We can define a matroid in terms of its rank function.

5.8 TYPES OF MATROIDS

5.8.1. Bipartite matroid

A bipartite matroid be a matroid, in which each cycle has an even number of elements.

5.8.2. Eulerian matroid

A matroid on a set E to be an Eulerian matroid if E can be written as a union of disjoint cycles.

5.8.3. Discrete matroids

At the other extreme is the discrete matroid on E, in which every subset of E is independent. The discrete matroid on E has only one base, E itself, and that the rank of any subset A is the number of elements in A.

5.8.4. Trivial matroids

Given any non-empty finite set E, we can define on it a matroid whose only indepenent set is the empty set ϕ. This matroid is the trivial matroid on E and has rank O.

5.8.5. Uniform matroids

The k-uniform matroid on E, whose bases are those subsets of E with exactly k elements, the trivial matroid on E is o-uniform and the discrete matroid is | E |-uniform. We note that the independent sets are those subsets of E with not more than k elements, and that the rank of any subset A is either | A | or k.

5.8.6. Isomorphic matroids

Two matroids M_1 and M_2 to be isomorphic if there is a one-one correspondence between their underlying sets E_1 and E_2 that preserves independence.

Thus, a set of elements of E_1 is independent in M_1 if and only if the corresponding set of elements of E_2 independent in M_2.

For example, the cycle matroids of the three graphs in Figure below are all isomorphic.

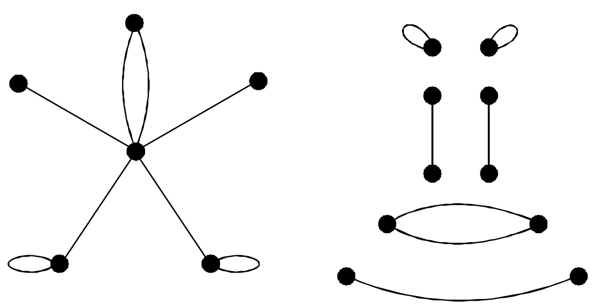


Fig. 5.1.

We note that, although matroid isomorphism preserves cycles, cutsets and the number of edges in a graph, it does not necessarily preserve connectedness, the number of vertices, or their degrees.

5.8.7. Graphic matroids

A matroid M(G) on the set of edges of a graph G by taking the cycles of G as the cycles of the matroid. M(G) is the cycle matroid of G and its rank function is the cutset rank ξ. It is natural to ask whether a given matroid M is the cycle matroid of some graph. In otherwords, does there exists a graph G such that M is isomorphic to M(G) ? Such matroids are called graphic matroids.

For example, the matroid M on the set {1, 2, 3} whose bases are {1, 2} and {1, 3} is a graphic matroid isomorphic to the cycle matroid of the graph in Figure below.

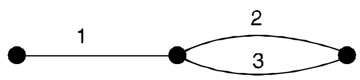


Fig. 5.2.

Note : A simple example of a non-graphic matroid is the 2-uniform matroid on a set of four elements.

5.8.8. Cographic matroids

Given a graph G, the cycle matroid M(G) is not the only matroid that can be defined on the set of edges of G. Because of the similarity between the properties of cycles and of cutsets in a graph, we can construct a matroid by taking the cutsets of G as cycles of the matroid.

We call it the cutset matroid of G, denoted by M*(G). We note that a set of edges of G is independent if and only if it contains no cutset of G. We call a matroid M cographic if there exists a graph G such that M is isomorphic to M*(G).

5.8.9. Planar matroids

A matroid that is both graphic and cographic is a planar matroid.

5.8.10. Transversal matroids

If E is a non-empty finite set and in F = (S_1, ... S_m) is a family of non-empty subsets of E, then the partial transversals of F can be taken as the independent sets of a matroid on E, denoted by M(F) or M(S_1, ..., S_m). Any matroid obtained in this way is a transversal matroid.

For example, the above graphic matroid M is a transversal matroid on the set {1, 2, 3}, since its independent sets are the partial transversal of the family F = (S_1, S_2), where S_1 = {1} and S_2 = {2, 3}.

We note that the rank of a subset of E is the size of the largest partial transversal contained in A.

Every transversal matroid is representable over some field, but is binary if and only if it is graphic.

5.8.11. The Fano matroids

The Fano matroid F is the matroid defined on the set E = {1, 2, 3, 4, 5, 6, 7}, whose bases are all those subsets of E with three elements, except {1, 2, 4}, {2, 3, 5}, {3, 4, 6}, {4, 5, 7}, {5, 6, 1}, {6, 7, 2} and {7, 1, 3}.

This matroid can be represented geometrically by Figure. below.

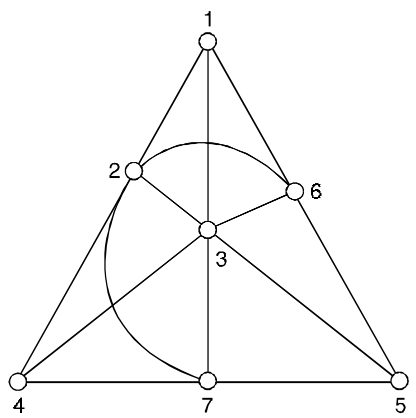


Fig. 5.3.

The bases are precisely those sets of three elements that do not lie on a line. It can be shown that F is binary and Eulerian, but is not graphic, cographic, transversal or regular.

5.8.12. Representable matroids

Given a matroid M on a set E, we say that M is representable over a field F if there exist a vector space V over F and a map ϕ from E to V, such that a subset of A of E is independent in M if and only if

ϕ is one-one on A and ϕ(A) is linearly independent in V. This amounts to saying that, if we ignore loops and parallel elements, then M is isomorphic to a vector matroid defined in some vector space over F.

For convenience, we say that M is a representable matroid if there exists some field F such that M is representable over F.

Some matroids are representable over every field (the regular matroids), some are representable over no field, and some are representable only over a restricted class of fields. The binary matroids representable over the field of integers modulo 2.

For example. If G is any graph, then its cycle matroids M(G) is a binary matroids. We associate with each edge of G the corresponding row of the incidence matrix of G, regarded as a vector with components 0 or 1.

We note that, if a set of edgs of G forms a cycle, then the sum (modulo 2) of the corresponding vectors is 0.

A binary matroid that is neither graphic nor cographic is the Fano matroid.

5.8.13. Restrictions and contractions

If M is a matroid defined on set E, and if A is a subset of E, then the restriction of M to A, denoted by M × A, is the matroid whose cycles are precisely those cycles of M that are contained in A. Similarly, the contraction of M to A, denoted by M . A, is the matroid whose cycles are the minimal members of the collection $\{C_i \cap A\}$, where C_i is a cycle of M.

A matroid obtained from M by restrictions and/or contractions is called a minor of M.

Theorem 5.1. *Every cocycle of a matroid intersects every base.*

Proof. Let C* be a cocycle of a matroid M suppose that there exists a base B of M with the property that C* ∩ B is empty.

Then C* is contained in E-B and so C* is a cycle of M* which is contained in a base of M*. This contradiction establishes the result.

Corollary. Every cycle of a matroid intersects every cobase.

Theorem 5.2. *The bases of M* are precisely the complements of the bases of M.*

Proof. We show that, if B* is a base of M* then E – B* is a base of M, the converse result is obtained by reversing the argument.

Since B* is independent in M*, $| B^* | = r^* (B^*)$ and hence $r(E – B^*) = r(E)$.

It remains only to prove that E – B* is independent in M. But this follows immediately from the fact that $r^*(B^*) = r^*(E)$.

Theorem 5.3. *A matroid is planar if and only if it is regular and contains no minor isomorphic to* $M(k_5)$, $M(k_{3,\,3})$ *or their duals.*

Theorem 5.4. *If G is a graph then M*(G) = (M(G))*.*

Proof. Since the cycles of M*(G) are the cutsets of G, we must check that C* is a cycle of (M(G))* if and only if C* is a cutset of G.

Suppose first that C* is a cutset of G. If C* is independent in (M(G))* then C* can be extended to a base B* of (M(G))* and so C* ∩ (E – B*) is empty. Since E – B* is a spanning forest of G.

Thus, C* is a dependent set in (M(G))*, and therefore contains a cycle of (M(G))*.

If, on the otherhand, D* is a cycle of (M(G))* then D* is not contained in any base of (M(G))*.

It follows that D* intersects every base of M(G) that is, every spanning forest of G.

Thus D* contains a cutset.

Theorem 5.5. *A matroid M is graphic if and only if it is binary and contains no minor isomorphic to M*(k_5), M*($k_{3,\,3}$), F or F*.*

Corollary. A matroid M is cographic if and only if it is binary and contains no minor isomorphic to M(k_5), M($k_{3,\,3}$), F or F*.

Theorem 5.6. *A graph G contains k edge-disjoint spanning forests if and only if for each subgraph H of G*

$$k(\xi(G) - \xi(H)) \le m(G) - m(H)$$

where m(H) and m(G) denote the number of edges of H and G respectively.

Theorem 5.7. *If M_1, ... M_k are matroids on a set E with rank functions r_1, ... r_k then the rank function r of $M_1 \cup ...... \cup M_k$ is given by*

$$r(X) = \min \{r_1(A) + ... + r_k(A) + | X - A |\},$$

where the minimum is taken over all subsets A of X.

Corollary. (1) Let M be a matroid, then M contains k disjoint bases if and only if for each subset A of E $kr(A) + | E - A | \ge kr(E)$.

Corollary. (2) Let M be a matroid then E can be expressed as the union of k independent sets if and only if for each subset A of E, $kr(A) \ge | A |$.

Theorem 5.8. *If G* is an abstract dual of a graph G then M(G)* is isomorphic to (M(G))*.*

Proof. Since G* is an abstract dual of G, there is a one-one correpondence between the edges of G and those of G* such that cycles in G corresponding to cutsets in G* and conversely.

It follows immediately that the cycles of M(G) correspond to the cycles of M(G)*.

Thus M(G*) is isomorphic to M*(G).

Corollary. If G* is a geometric dual of a connected plane graph G, then M(G*) is isomorphic to (M(G))*.

Theorem 5.9. *Let E be a non empty finite set, and let F = (S_1, ..., S_m) and G = (T_1, ..., T_m) be two families of non empty subsets of E. Then F and G have a common transversal if and only if for all subsets A and B of {1, 2, ..., m},*

$$\left| \left(\bigcup_{i \in A} S_i \right) \cap \left(\bigcup_{j \in A} T_j \right) \right| \geqq | A | + | B | - m.$$

Proof. Let M be the matroid whose independent sets are precisely the partial transversals of the family F. Then F and G have a common transversal if and only if G has an independent transversal.

This is so if and only if the union of any k of the sets T_i contains an independent set of size atleast k, for $1 \le k \le M$, that is, if and only if the union of any k of the sets T_i contains a partial transversal of F of size k.

Theorem 5.10. *A matroid consists of a non-empty finite set E and an integer valued function r defined on the set of subsets of E satisfying*

r(i) $0 \le r(A) \le |A|$ *for each subset A of E.*

r(ii) if $A \subseteq B \subseteq E$, *then* $r(A) \le r(B)$

r(iii) for any $A, B \subseteq E$, $r(A \cup B) + r(A \cap B) \le r(A) + r(B)$.

Proof. We assume that M is a matroid defined in terms of its independent sets. We wish to prove properties $r(i) - r(iii)$. Clearly, properties $r(i)$ and $r(ii)$ are trivial. To prove $r(iii)$, we let X be a base (a maximal independent subset) of $A \cap B$. Since X is an independent subset of A, X can be extended to a base Y of A, and then to a base Z of $A \cup B$. Since $X \cup (Z - Y)$ is clearly an independent subset of B, it follows that

$$\begin{aligned} r(B) &\ge r(X \cup (Z - Y)) \\ &= |X| + |Z| - |Y| \\ &= r(A \cap B) + r(A \cup B) - r(A) \text{ as requuired.} \end{aligned}$$

Conversely, let $M = (E, r)$ be a matroid defined in terms of a rank function r, and define a subset A of E to be independent if and only if $r(A) = |A|$. It is then simple to prove property I(i). To prove I(ii), let I and J be independent sets with $|J| > |I|$ and suppose that $r(I \cup \{e\}) = k$ for each element e that lies in J but not in I. If e and f are two such elements then

$$e(I \cup \{e\} \cup \{f\}) \le r(I \cup \{e\}) + r(I \cup \{f\}) - r(I) = k$$

If follows that $r(I \cup \{e\} \cup \{f\}) = k$. We now continue this procedure, adding one new element of J at a time. Since at each stage the rank k we conclude that $r(I \cup J) = k$ and hence (by $r(ii)$) that $r(J) \le k$, which is a contradiction. It follows that there exists an element f that lies in J but not in I, such that

$$r(I \cup \{f\}) = k + 1.$$

Theorem 5.11. *Let M be a matroid on a set E, and let* $F = (S_1, \ldots, S_m)$ *be a family of non-empty subsets of E. Then F has an independent transversal if and only if the union of any k of the subsets* S_i *contains an independent set of size at least k, for* $1 \le k \le m$.

Proof. We show that if one of the subsets (S_1, say) contains more than one element then we can remove an element from S_1 without altering the condition. By repeating this procedure, we eventually reduce the problem to the case in which each subset contains only one element and the proof is then trivial.

It remains only to show the validity of this 'reduction procedure', so, suppose that S_1 contains elements x and y, the removal of either of which in validates the condition.

Then there are subsets A and B of $\{2. 3, \ldots, m\}$ with the property that $r(P) \le |A|$ and $r(Q) \le |B|$, where

$$P = \bigcup_{j \in A} S_j \cup (S_1 - \{x\}) \text{ and } Q = \bigcup_{j \in B} S_j \cup (S_1 - \{y\})$$

Then $\quad r(P \cup Q) = r\left(\bigcup_{j \in A \cup B} S_j \cup S_1\right)$ and $r(P \cup Q) \ge r\left(\bigcup_{j \in A \cap B} S_j\right)$

The required contradiction now follows, since

$$\begin{aligned} |A| + |B| &\geq r(P) + r(Q) \\ &\geq r(P \cup Q) + r(P \cap Q) \\ &\geq \left| \bigcup_{j \in A \cup B} S_j \cup S_1 \right| + \left| \bigcup_{j \in A \cap B} S_j \right| \\ &\geq (|A \cup B| + 1) + |A \cap B| \\ &= |A| + |B| + 1. \end{aligned}$$

Corollary. With the above notation, F has an independent partial transveral of size t if and only if the union of any k of the subsets S_i contains an independent set of size at least $k + t - m$.

5.9 TRANSVERSAL THEORY

5.9.1. Transversal

If E is a non empty finite set, and if F = (S_1, ... S_m) is a family of (not necessarily distinct) non-empty subsets of E, then a transversal of F is a set of m distinct elements of E, one chosen from each set S_i.

Suppose that E = {1, 2, 3, 4, 5, 6} and

Let $S_1 = S_2 = \{1, 2\}$, $S_3 = S_4 = \{2, 4\}$, $S_5 = \{1, 4, 5, 6\}$

Then it is impossible to find five distinct elements of E, one from each subset S_i ;

In otherwords, the family F = (S_1, ..., S_5) has no transversal.

The subfamily F′ = (S_1, S_2, S_3, S_4, S_5) has a transversal.

For example, {1, 2, 3, 4, 5}.

5.9.2. Partial transversal

A transversal of a subfamily of F a partial transversal of F.

For example, {1, 2, 3, 4}, here F has several partial transversal, such as, {1, 2, 3, 6}, {2, 3, 6}, {1, 5}, and ϕ.

We note that any subset of a partial transversal is a partial transversal.

Note : A given family of subsets of a set has a transversal. The connection between this problem and the marriage problem is easily seen by taking E to be the set of boys, and S_i to be the set of boys known by girl g_i, for $1 \leq i \leq m$.

A transversal in this case is then simply a set of m boys, one corresponding to, and known by, each girl.

5.9.3. Marriage problem

If there is a finite set of girls, each of whom knows several boys, under what conditions can all the girls marry the boys in such a way that each girl marries a boy she knows ?

For example. If there are four girls {g_1, g_2, g_3, g_4} and five boys {b_1, b_2, b_3, b_4, b_5} and the friendships are as shown in Figure 5.4(a) and (b), below

girls	*boys known by girl*		
g_1	b_1	b_4	b_5
g_2	b_1		
g_3	b_2	b_3	b_4
g_4	b_2	b_4	

Fig. 5.4. (*a*)

then a possible solution is for g_1 to marry b_4, g_2 to marry b_1, g_3 to marry b_3, and g_4 to marry b_2.

This problem can be represented graphically by taking G to be the bipartite graph in which the vertex set is divided into two disjoint sets V_1 and V_2, corresponding to the girls and boys, and where each edge joins a girl to a boy she knows. Figure 5.4(*b*) shows the graph G corresponding to the situation of Figure 5.4(*a*).

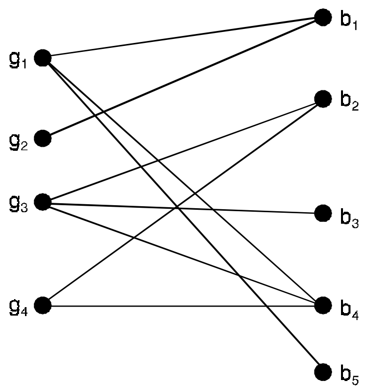


Fig. 5.4. (*b*)

In graph-theoretic form

If G = G(V_1, V_2) is a bipartite graph, when does there exist a complete matching from V_1 to V_2 in G ?

5.9.4. Marriage Condition

A "matrimonial terminology", we note that, for the solution of the marriage problem, every k girls must know collectively at least k boys, for all integers k satisfying $1 \le k \le m$, where m denotes the total number of girls. We refer to this condition as the marriage condition.

It is a necessary condition and it turns out to be sufficient.

5.9.5. Common transversals

If E is a non-empty finite set and F = (S_1, ..., S_m) and G = (T_1, ..., T_m) are two families of non-empty subsets of E, it is of interest to know when there exists a common transversal for F and G.

i.e., *a* set of m distinct elements of E that forms a transversal of both F and G.

For example, In time tabling problems, E may be the set of times at which lectures can by given, the sets S_i may be the times that m given professors are willing to lecture, and the sets T_i may be the times that m lecture rooms are available. Then the finding of a common transversal of F and G enables us to assign to each professor an available lecture root at a suitable time.

5.9.6. Latin squares

An $m \times n$ latin rectangle is an $m \times n$ matrix M = (m_{ij}) whose entries are integers satisfying

(i) $1 \le m_{ij} \le n$ (ii) no two entries in any row or in any column are equal.

We note that (i) and (ii) imply that $n \le n$.

If $m = n$, then the latin rectanle is a latin square.

For example, Figure 5.5(a) and 5.5(b) shows a 3 × 5 latin rectangle and a 5 × 5 latin square.

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 1 & 5 & 3 \\ 2 & 5 & 2 & 1 & 4 \end{bmatrix} \qquad \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 1 & 5 & 3 \\ 3 & 5 & 2 & 1 & 4 \\ 4 & 3 & 5 & 2 & 1 \\ 5 & 1 & 4 & 3 & 2 \end{bmatrix}$$

Fig. 5.5(a) **Fig. 5.5(b)**

5.9.7. (0, 1) – matrices

The way of studying transversals of a family F = $(S_1, ..., S_m)$ of non-empty subsets of a set E = $\{e_1, ..., e_n\}$ is to study the incidence matrix of the family, the $m \times n$ matrix A = (a_{ij}) in which $a_{ij} = 1$ if $e_j \in S_i$ and $a_{ij} = 0$ otherwise. We call such a matrix, in which each entry is 0 or 1, a(0, 1) – matrix.

Note : If the term rank of A is the largest number of 1s of A, no two of which lie in the same row or column, then F has a transversal if and only if the term rank of A is m. Moreover, the term rank of A is precisely the number of elements in a partial transversal of largest possible size.

5.9.8. Edge-disjoint paths

The number of paths connecting two given vertices v and w in a graph G. We may ask for the maximum number of paths from v to w, no two of which have an edge in common, such paths are called edge-disjoint paths.

5.9.9. Vertex-disjoint paths

The number of paths connecting two given vertices v and w in a graph G. We may ask for the maximum number of paths from v to w, no two of which have a vertex in common, such paths are called vertex-disjoint paths.

For example, in the graph of Fig. (5.6), there are four edge-disjoint paths are two vertex-disjoint ones.

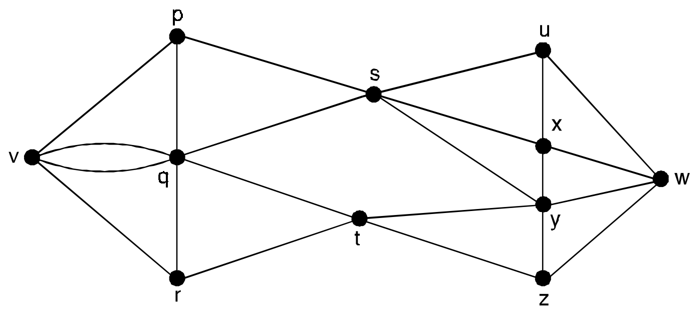


Fig. 5.6.

5.9.10. *vw*-disconnecting set

Assume that G is a connected graph and that v and w are distinct vertices of G. A *vw*-disconnecting set of G is a set E of edges of G such that each path from v to w includes an edge of E. We note that a *vw*-disconnecting set is a disconnecting set of G.

5.9.11. *vw*-separating set

Assume that G is a connected graph and that v and w are distinct vertices of G. A *vw*-separating set of G is a set S of vertices, other than v or w, such that each path from v to w passes through a vertex of S.

In Fig, (5.6), the sets $E_1 = \{ps, qs, ty, tz\}$ and $E_2 = \{uw, xw, yw, zw\}$ are *vw*-disconnecting sets, and $V_1 = \{s, t\}$ and $V_2 = \{p, q, y, z\}$ are *vw*-separating sets.

Theorem 5.12. *Let E be a non-empty finite set, and let F = (S_1, ..., S_m) be a family of non-empty subsets of E. Then F has a transversal if and only if the union of any k of the subsets S_i contains at least k elements ($1 \le k \le m$).*

Proof. The necessity of the condition is clear. To prove the sufficiency.

We show that if one of the subsets (S_1, say) contains more than one element, then we can remove an element from S_1 without altering the condition.

By repeating this procedure, we eventually reduce the problem to the case in which each subset contains only one element, and the proof is then trivial.

In remains only to show the validity of this 'reduction procedure'.

Suppose that S_1 contains elements x and y, the removal of either of which invalidates the condition.

Then there are subsets A and B of {2, 3, ..., m} with the property that $|P| \le |A|$ and $|Q| \le |B|$

Where $$P = \bigcup_{j \in A} S_j \cup (S_1 - \{x\}) \text{ and } Q = \bigcup_{j \in A} S_j \cup (S_1 - \{y\})$$

Then $$|P \cup Q| = \left| \bigcup_{j \in A \cup B} S_j \cup S_1 \right| \text{ and } |P \cup Q| \ge \left| \bigcup_{j \in A \cap B} S_j \right|$$

The required contradiction now follows, since

$$\begin{aligned} |A| + |B| &\geqq |P| + |Q| \\ &= |P \cup Q| + |P \cap Q| \\ &\geq \left| \bigcup_{j \in A \cup B} S_j \cup S_1 \right| + \left| \bigcup_{j \in A \cap B} S_j \right| \\ &\geqq (|A \cup B| + 1) + |A \cap B| \\ &= |A| + |B| + 1. \end{aligned}$$

Corollary 1. If E and F are as before, then F has a partial transversal of size t if and only if the union of any k of the subsets S_i contains at least $k + t - m$ elements.

Remark : On applying theorem, to the family $F' = (S_1 \cup D \text{ .....}, S_m \cup D)$ where D is any set disjoint from E and containing $m - t$ elements.

We note that F has a partial transversal of size t if and only if F′ has a transversal.

Corollary 2. If E and F are as before, and if X is any subset of E, then X contains a partial transversal of F of size t if and only if, for each subset A of $\{1, \text{......}, m\}$. $\left| \left(\bigcup_{j \in A} S_j \right) \cap X \right| \geq |A| + t - m$.

Theorem 5.13. *Hall's theorem*

A necessary and sufficient condition for a solution of the marriage problem is that each set of k girls collectively knows at least k boys, for $1 \leq k \leq m$.

Proof. The condition is necessary. To prove that it is sufficient, we use induction on m.

Assume that the theorem is true if the number of girls is less than m.

We note that the theorem is true if $m = 1$.

Suppose that there are m girls. There are two cases to consider.

Case (*i*) If every k girls (where $k < m$) collectively know at least $k + 1$ boys, so that the condition is always true 'with one boy to spare', then we take any girl and marry her to any boy she knows.

The original condition then remains true for the other $m - 1$ girls, who can be married by induction, completing the proof in this case.

Case (*ii*) If now there is a set of k girls ($k < m$) who collectively know exactly k boys, than these k girls can be married by induction to the k boys, leaving $m - k$ girls still to be married. But any collection of h of these $m - k$ girls, for $h \leq m - k$ must know at least h of the remaining boys, since otherwise these h girls, together with the above collection of k girls, would collectively know fewer than $h + k$ boys, contrary to our assumptuon.

It follows that the original condition applies to the $m - k$ girls. They can therefore be married by induction in such a way that everyone is happy and the proof is complete.

Corollary. Let $G = G(V_1, V_2)$ be a bipartite graph, and for each subset A of V_1, let $\phi(A)$ be the set of vertices of V_2 that are adjacent to at least one vertex of A. Then a complete matching from V_1 to V_2 exists if and only if $|A| \leqq |\phi(A)|$, for each subset A of V_1.

Theorem 5.14. *Let E be a non-empty finite set, and let F = (S_1, ..., S_m) and G = (T_1, ..., T_m) be two families of non-empty subsets of E. Then F and G have a common transversal if and only if, for all subsets A and B of {1, 2, ..., m},* $P = \left|\left(\bigcup_{i\in A} S_i\right)\cap\left(\bigcup_{j\in B} T_j\right)\right| \geq |A| + |B| - m.$

Proof. Consider the family $x = \{X_i\}$ of subsets of $E \cup \{1, ..., m\}$.

Assuming that E and {1, ..., m} are disjoint, where the indexing set is also $E \cup \{1, ..., m\}$ and where

$$X_i = S_i \text{ if } i \in \{1, ..., m\} \text{ and } X_i = \{i\} \cup \{j : j \in T_j\} \quad \text{if } i \in E.$$

It is not difficult to verify that F and G have a common transversal if and only if x has a transversal. The result follows on applying Hall's theorem to the family x.

Theorem 5.15. *Let M be an m × n latin rectangle with m < n. Then, M can be extended to a latin square by the addition of n – m new rows.*

Proof. We prove that M can be extended to an $(m + 1) \times n$ latin rectangle. By repeating the procedure involved, we eventually obtain a latin square.

Let E = {1, 2, ..., n} and F = (S_1, ..., S_n), where S_i is the set consisting of those elements of E that do not occur in the ith column of M.

If we can prove that F has a transversal, then the proof is complete, since the elements in this transversal form the additional row.

By Hall's theorem, it is sufficient to show that the union of any k of the S_i contains at least k distinct elements.

But this is obvious, since such a union contains $(n - m)\, k$ elements altogether, including repetitions, and if there were fewer than k distinct elements, then at least one of them would have to appear more than $n - m$ times.

Since each element occurs exactly $n - m$ times, we have the required contradiction.

Theorem 5.16. *König-Egervary theorem*

The term rank of a (0, 1) – matrix A is equal to the minimum number μ of rows and columns that together contain all the 1s of A.

Proof. It is clear that the term rank cannot exceed μ. To prove equality, we can suppose that all the 1s of A are contained in r rows and S columns, where $r + s = \mu$, and that the order of the rows and columns is such that A contains, in the bottom left-hand corner, an $(m - r) \times (n - s)$ such matrix consisting entirely of 0s (See Fig. (5.7)).

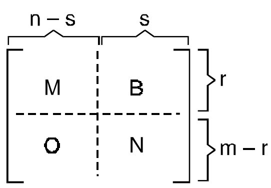


Fig. 5.7.

If $i < r$, let S_i be the set of integers $j \le n - s$ such that $a_{ij} = 1$. It is simple to check that the union of any k of the S_i contains at least k integers, and hence that the family F = (S_1, ..., S_r) has a transversal. It follows that the maxtrix M of A contains a set of r 1s, no two of which lie in the same row or column.

Similarly, the maxtrix N contains a set of S 1s with the same row or column. This shows that μ cannot exceed the term rank, as required.

Remark. Consider the matrix of Fig. 5.7(a) below, which is the incidence matrix of the family F = (S_1, S_2, S_3, S_4, S_5) of subsets of E = {1, 2, 3, 4, 5, 6}, where $S_1 = S_2$ = {1, 2}, $S_3 = S_4$ = {2, 3}, S_5 = {1, 4, 5, 6}.

Clearly the term rank and μ are both 4.

$$\begin{array}{c} \\ S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Fig. 5.7(*a*)

Theorem 5.17. *In any binary matrix, the maximum number of independent unit elements equals the minimum number of lines which cover all the units.*

$$M = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \qquad M' = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Theorem 5.18. *There exists a system of distinct representatives for a family of sets S_1, S_2, ..., S_m if and only if the union of any k of these sets contains at least k elements, for all k from 1 to m.*

Proof. The necessity is immediate. For the sufficiency we first prove that if the collection {S_i} satisfies the stated conditions and $|S_m| \ge 2$, then there is an element e in S_m such that the collection of sets S_1, S_2, ..., S_{m-1}, $S_m - \{e\}$ also satisfies the conditions.

Suppose this is not the case. Then there are elements e and f in S_m and subsets J and k of {1, 2, ..., $m - 1$}

such that $$\left|\left(\bigcup_{i \in J} S_i\right) \cup (S_m - \{e\})\right| < |J| + 1 \text{ and } \cup \left|\left(\bigcup_{i \in K} S_i\right) \cup (S_m - \{f\})\right| < |K| + 1$$

But then $$|J| + |K| \ge \left|\left(\bigcup_{J} S_i\right) \cup (S_m - \{e\})\right| + \left|\left(\bigcup_{K} S_i\right) \cup (S_m - \{f\})\right|$$

$$\geq \left| \left(\bigcup_{J \cup K} S_i \right) \cup S_m \right| + \left| \bigcup_{J \cap K} S_i \right|$$

$$\geq | J \cup K | + 1 + | J \cap K | > | J | + | K |$$

which is a contradiction.

Theorem 5.19. *If any finite lattice, the maximum number of incomparable elements equals the minimum number of chains which include all the elements.*

Theorem 5.20. *A graph with atleast 2n points is n-connected if and only if for any two disjoint sets V_1 and V_2 of n points each, there exist n disjoint paths joining these two sets of points.*

Proof. To show the sufficiency of the condition, we form the graph G′ from G by adding two new points W_1 and W_2 with W_i adjacent to exactly the points of V_i, $i = 1, 2$. (see Fig. 5.8 below)

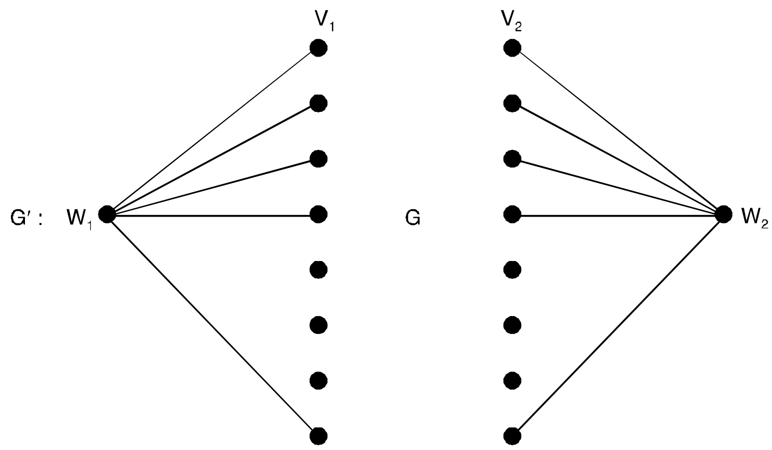


Fig. 5.8. Construction of G′.

Since G is *n*-connected, so is G′, there are *n* disjoint paths joining W_1 and W_2.

The restrictions of these paths to G are clearly the *n* disjoint $V_1 - V_2$ paths we need.

To prove the other 'half', let S be a set of at least $n - 1$ points which separates G into G_1 and G_2, with points sets V_1' and V_2' respectively.

Then, since $| V_1' | \geq 1$, $| V_2' | \geq 1$ and $| V_1' | + | V_2' | + | S | = | V | \geq 2n$, there is a partition of S into two disjoint subsets S_1 and S_2 such that $| V_1' \cup S_1 | \geq n$ and $| V_2' \cup S_2 | \geq n$.

Picking any *n*-subsets V_1 of $V_1' \cup S_1$ and V_2 of $V_2' \cup S_2$, we have two disjoint sets of *n* points each.

Every path joining V_1 and V_2 must contain a point of S, and since we know there are *n* disjoint $V_1 - V_2$ paths, we see that $| S | \geq n$, and G is *n*-connected.

Theorem 5.21. *For any two points of a graph, the maximum number of line disjoint paths joining them equals the minimum number of lines which separate them.*

Theorem 5.22. *The maximum number of arc-disjoint paths from a vertex v to a vertex w in a digraph is equal to the minimum number of arcs in a vw-disconnecting set.*

Theorem 5.23. *Menger's theorem*

The maximum number of edge-disjoint paths connecting two distinct vertices v and w of a connected graph is equal to the minimum number of edges in a vw-disconnected set.

Proof. The maximum number of edge-disjoint paths connecting v and w cannot exceed the minimum number of edges in a vw-disconnecting set.

We use induction on the number of edges of the graph G to prove that these numbers are equal.

Suppose that the number of edges of G is m, and that the theorem is true for all graphs with fewer than m edges. There are two cases to consider.

Case (*i*). Suppose first that there exists a vw-disconnecting set E of minimum size k, such that not all of its edges are incident to v, and not all are incident to w. For example, in Fig. 5.6, the above set E_1 is such a vw-disconnecting set.

The removal from G of the edges in E leaves two disjoint subgraphs V and W containg v and w, respectively.

We now define two new graphs G_1 and G_2 are follows : G_1 is obtained from G by contracting every edge of V, that is, by shrinking V down to v, and G_2 is obtained by similarly contracting every edge of W ; the graphs G_1 and G_2 obtained from Fig. 5.6 are shown in Fig. 5.9 below, with dashed lines denoting the edges of E_1.

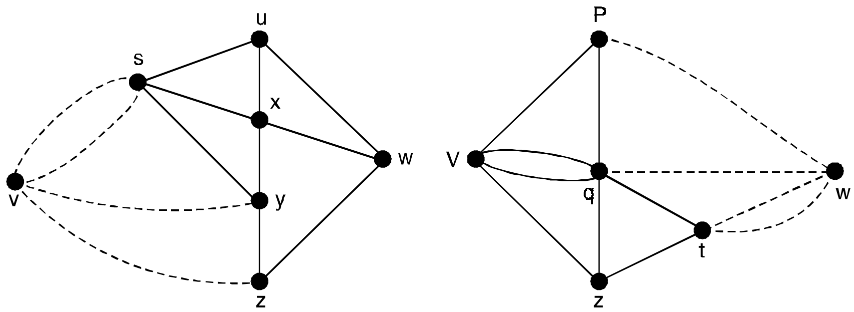


Fig. 5.9.

Since G_1 and G_2 have fewer edges than G, and since E is a vw-disconnecting set of minimum size for both G_1 and G_2, the induction hypothesis gives us k edge-disjoint paths in G_1 from v to w, and similarly for G_2,

The required k edge-disjoint paths in G are obtained by combining these paths in the obvious way.

Case (*ii*). Now suppose that each vw-disconnecting set of minimum size k consists only of edges that are all incident to v or all incident to w.

For example, in Fig. 5.6, the set E_2 is such a vw-disconnecting set.

We can assume without loss of generality that each edge of G is contained in a vw-disconnecting set of size k, since otherwise its removal would not affect the value of k and we could use the induction hypothesis to obtain k edge-disjoint paths.

If P is any path from v to w, then P must consist of either one or two edges, and can thus contain atmost one edge of any vw-disconnecting set of size k.

By removing from G the edges of P, we obtain a graph with at least $k - 1$ edge-disjoint paths, by the induction hypothesis. These paths, together with P, give the required k paths in G.

Theorem 5.24. *Menger's*

The maximum number of vertex-disjoint paths connecting two distinct non-adjacent vertices v and w of a graph is equal to the minimum number of vertices in a vw-separating set.

Corollary (1) A graph G is k-edge-connected if and only if any two distinct vertices of G are connected by atleast k edge-disjoint paths.

Corollary (2) A graph G with atleast $k + 1$ vertices is k-connected if and only if any two distinct vertices of G are connected by atleast k vertex-disjoint paths.

Theorem 5.25. *Menger's theorem implies Hall's theorem*.

Proof. Let $G = G(V_1, V_2)$ be a bipartite graph.

We must prove that, if $|A| \le |\phi(A)|$ for each subset A of V_1, then there is a complete matching from V_1 to V_2.

To do this, we apply the vertex form of Menger's theorem to the graph obtained by adjoining to G a vertex v adjacent to every vertex in V_1 and a vertex w adjacent to every vertex in V_2 (see Fig. 5.10 below).

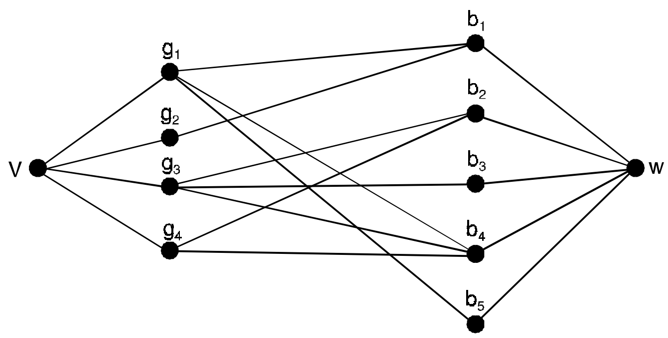


Fig. 5.10.

Since a complete matching from V_1 to V_2 exist if and only if the number of vertex-disjoint paths from v to w is equal to the number of vertices in V_1 (= k, say), it is enough to show that every vw-separating set has atleast k vertices.

So, let S be a vw-separating set consisting of a subset A of V_1 and a subset B of V_2.

Since $A \cup B$ is a vw-separating set, no edge can join a vertex of $V_1 - A$ to a vertex of $V_2 - B$ and hence $\phi(V_1 - A) \subseteq B$.

It follows that $|V_1 - A| \le |\phi(V_1 - A)| \le |B|$ and so $|S| = |A| + |B| \ge |V_1| = k$, as required.

Problem Set 5.1

1. Prove that a matroid M is a transversal matroid if and only if M can be expressed as the union of matroids of rank 1.

2. Let E = {*a*, *b*, *c*, *d*, *e*}. Find matroids on E for which

(*i*) E is the only base

(*ii*) the empty set ϕ is the only base

(*iii*) the bases are those subsets of E containing exactly three elements.

For each matroid, write down the independent sets, the cycles (if there are any) and the rank function.

3. A matroid consists of a non-empty finite set E and an integer valued function *r* defined on the set of subsets of E, satisfying

$r(i)$ $0 \le r(A) \le |A|$ for each subset A of E

$r(ii)$ if $A \subseteq B \subseteq E$ then $r(A) \le r(B)$

$r(iii)$ for any $A, B \subseteq E$, $r(A \cup B) + r(A \cap B) \le r(A) + r(B)$.

4. Let M be the matroid on the set E = {*a*, *b*, *c*, *d*} whose bases are {*a*, *b*}, {*a*, *c*}, {*a*, *d*}, {*b*, *c*}, {*b*, *d*} and {*c*, *d*}. Write down the cycles of M, and deduce that there is no graph G with M as its cycle matroid.

5. Show how the definition of a fundamental system of cycles in a graph can be extended to matroids.

6. Let E = {1, 2, 3, 4, 5, 6} and $F = \{S_1, S_2, S_3, S_4, S_5)$ where $S_1 = S_2 = \{1, 2\}$, $S_3 = S_4 = \{2, 3\}$, $S_5 = \{1, 4, 5, 6\}$

(*i*) Write down the partial transversals of F and check that they form the independent sets of a matroid on E.

(*ii*) Write down the bases and cycles of this matroid.

7. Let G_1 and G_2 be the graphs shown in Figure 5.11 below, write down the bases, cycles and independent sets of the cycle matroids $M(G_1)$ and $M(G_2)$.

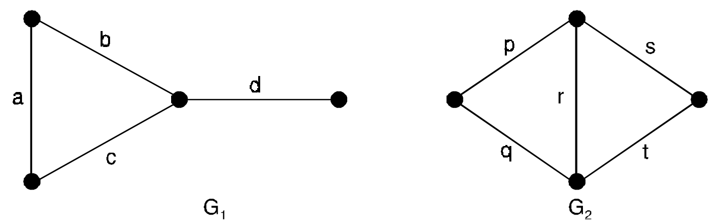


Fig. 5.11.

8. Prove that if M satisfies any of the following properties then so does any minor of M :

(*i*) graphic (*ii*) cographic (*iii*) binary (*iv*) regular.

9. Let E = {*a*, *b*, *c*}. Show that there are (up to isomorphism) exactly eight matroids on E, and list their bases independent sets and cycles.

10. Show that every uniform matroid is a transversal matroid.

11. Prove that every cocycle of a matroid intersects every base.

12. Prove that every cycle of a matroid intersects every cobase.

13. Let M be a binary matroid on a set E

(*i*) prove that if M is an Eulerian matroid then M* is bipartite

(*ii*) use induction on | E | to prove the converse result.

(*iii*) By considering the 5-uniform matroid on a set of 11 elements, show that the word 'binary' cannot be omitted.

14. Show that the contraction matroid M, A is the matroid whose cocycles are those cocycles of M that are contained in A.

15. Prove that a matroid M is graphic if and only if it is binary and contains no minor isomorphic to $M^*(K_5)$, $M^*(K_{3,\,3})$, for F*.

16. If G* is an abstract dual of a graph G, then prove that M(G*) is isomorphic to (M(G))*.

17. If G* is a geometric dual of a connected plane graph G then prove that M(G*) is isomorphic to (M(G))*.

18. Prove that a matroid is planar if and only if it is a regular and contains no minor isomorphic to $M(K_5)$, $M(K_{3,\,3})$ or their duals.

19. Let M be a matroid then prove that M contains k disjoint bases if and only if, for each subset of E. $Kr(A) + |E - A| \geq Kr(E)$.

20. Let M be a matroid on a set E and let $F = (S_1, ..., S_m)$ be a family of non-empty subsets of E. Then prove that F has an independent transversal if and only if the union of only k of the susbets S_i contains an independent set of size atleast k, for $1 \leq k \leq m$.

21. If M_1, ..., M_k are matroids on a set E with rank functions r_1, ..., r_k then prove that the rank function r of $M_1 \cup ...... \cup M_k$ is given by

$$r(X) = \min \{r_1(A) + ... + r_k(A) + |X - A|\}$$

where the minimum is taken over all subsets A of X.

22. Let M be a matroid, then prove that E can be expressed as the union of k independent sets if and only if for each subset A of E, $Kr(A) \geq |A|$.

23. What are the cocycles and cobases of

(*i*) the 3-uniform matroid on a set of 9 elements ?

(*ii*) the cycle matroids of the graphs in Figure 12(*a*) below

(*iii*) the cycle matroid of the graph in Figure 12(*b*) below

(*iv*) the Fano matroid.

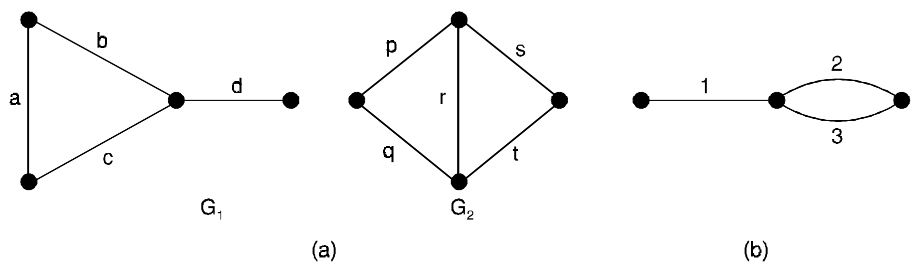


Fig. 5.12.

24. Show that the dual of a discrete matroid is a trivial matroid.

25. What is the dual of the *k*-uniform matroid on a set of *n* elements ?

26. Let E be a set of *n* elements. Show that upto isomorphism

(*i*) the number of matroids on E is at most 2^{2n}

(*ii*) the number of transversal matroids on E is at most 2^{n^2}.

27. Let E = {*a*, *b*}. Show that there are (upto isomorphism) exactly four matroids on E, and list their bases, independent sets and cycles.

28. State and prove a matroid analogue of the greedy algorithm.

29. Let E be the set {1, 2, ... 50}. How many distinct transversals has the family ({1, 2}, {2, 3}, {3, 4}, ... {50, 1}).

30. Prove that the term rank of *a*(0, 1)-matrix A is equal to the minimum number μ of rows and columns that together contain all the 1s of A.

31. Let M be a an $m \times n$ latin rectangle with $m < n$, then prove that M can be extended to a latin square by the addition of $n - m$ new rows.

32. Verify the König-Egerváry theorem for the following matrices :

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

33. Find two ways of completing the following latin rectangle to a 5×5 latin square $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 3 & 1 & 2 & 4 \end{pmatrix}$.

34. Give an example of a 5×8 latin rectangle and a 6×6 latin square.

CHAPTER

6 Enumeration and Groups

INTRODUCTION

Cayley's (1857) classic paper, a great deal of work has been done on enumeration (also called counting) of different types of graphs, and the results have been applied in solving some practical problems.

The Pioneers is graphical enumeration theory were **Cayley, Redfield and Pólya.** Graphical enumeration methods in current use were anticipated in the unique paper by Redfield pubished in 1927 Enumerative techniques will be developed and used for counting certain types of graphs. A thorough exposition of **Pólya's counting theorem,** the most powerful tool in graph enumeration.

6.1 TYPES OF ENUMERATION

Type 1. Counting the number of different graphs (or digraphs) of a particular kind.

For example, all connected, simple graphs with eight vertices and two circuits.

Type 2. Counting the number of subgraphs of a particular type in a given graph G, such as the number of edge-disjoint paths of length k between vertices a and b in G.

In problems of type 1 the word 'different' is of utmost importance and must be clearly understood. If the graphs are labeled, *i.e.*, each vertex is assigned a name distinct from all others, all graphs are counted on the otherhand, in the case of unlabeled graphs the word 'different' means non-isomorphic, and each set of isomorphic graphs is counted as one.

For example, let us consider the problem of constructing all simple graphs with n vertices and e edges. There are $\frac{n(n-1)}{2}$ unordered pairs of vertices. If we regard the vertices as distinguishable from one another *i.e.,* labeled graphs, there are $\begin{pmatrix} \frac{n(n-1)}{2} \\ e \end{pmatrix}$ ways of selecting e edges to form the graph.

Thus $\begin{pmatrix} \frac{n(n-1)}{2} \\ e \end{pmatrix}$ gives the number of simple labeled graphs with n vertices and e edges.

In the problems of type 2, usually involves a matrix representation of graph G and manipulations of this matrix. Such problems, although after encountered in practical applications, are not as varied and interesting as those in the first category.

6.2 LABELED GRAPHS

All of the labeled graphs with three points are shown in Figure 6.1 below. We see that the 4 different graphs with 3 points become 8 different labeled graphs. To obtain the number of labeled graphs with P points, we need only observe that each of the $\binom{P}{2}$ possible lines is either present or absent.

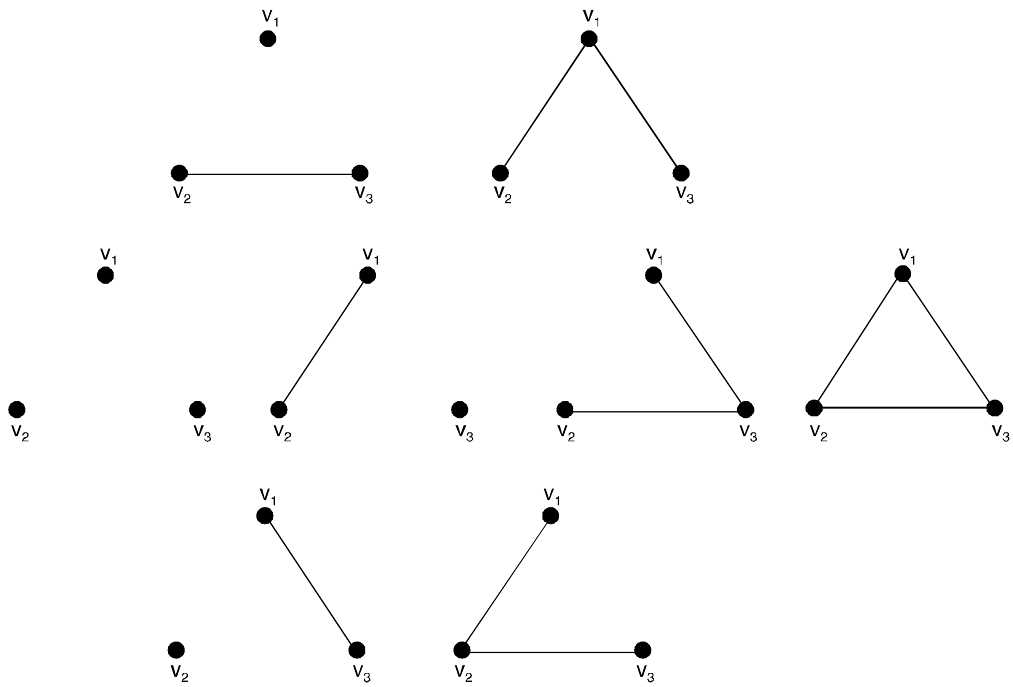


Fig. 6.1. The labeled graphs with 3 points.

6.3 COUNTING LABELED TREES

Expression $\left(\begin{array}{c}\dfrac{n(n-1)}{2}\\ e\end{array}\right)$ can be used to obtain the number of simple labeled graphs of n vertices and $n - 1$ edges. Some of these are going to be trees and others will be unconnected graphs with circuits.

For example, In Figure 6.2 below are all the 16 labeled trees with 4 points. The labels on these trees are understood to be as in the first and last trees shown.

We note that among these 16 labeled trees, 12 are isomorphic to the path P_4 and 4 to $k_{1,\,3}$.

The order of $\Gamma(P_4)$ is 2 and that of $\Gamma(k_{1,\,3})$ is 6.

We observe that since P = 4 here, we have

$$12 = \frac{4!}{|\Gamma(P_4)|} \text{ and } 4 = \frac{4!}{|\Gamma(k_{1,3})|}$$

The expected generalization of these two observations holds not only for trees, but also for graphs, digraphs, and so forth.

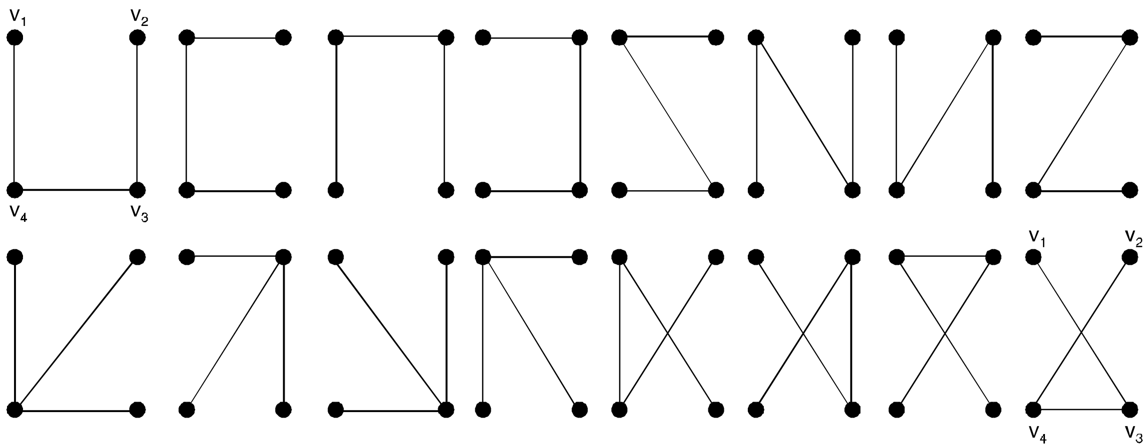


Fig. 6.2. The labeled trees with 4 points.

6.4 ROOTED LABELED TREES

In a rooted graph one vertex is marked as the root. For each of the n^{n-2} labeled trees we have n rooted labeled trees, because any of the n vertices can be made a root. Therefore, the number of different rooted labeled trees with n vertices is n^{n-1}.

6.5 ENUMERATION OF GRAPHS

To obtain the polynomial $g_P(x)$ which enumerates graphs with a given number P of points. Let g_{pq} be the number of (p, q) graphs and let $g_P(x) = \sum_q g_{pq} x^q$, all graphs with 4 points ; $g_4(x) = 1 + x + 2x^2 + 3x^3 + 2x^4 + x^5 + x^6$.

6.6 ENUMERATION OF TREES

To find the number of trees it is necessary to start by counting rooted trees. A rooted tree has one point, its root, distinguished from the others. Let T_P be the number of rooted trees with P points.

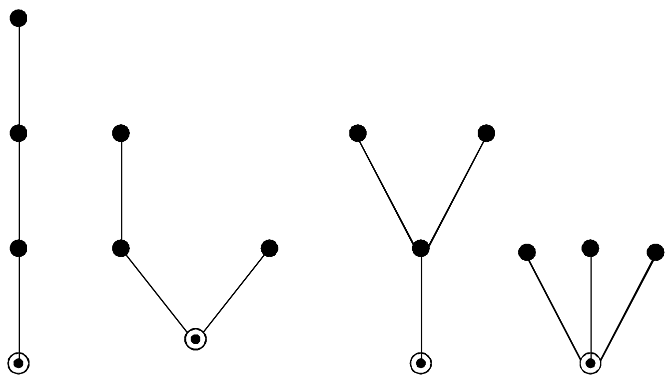


Fig. 6.3.

From figure 6.3 above, in which the root of each tree is visibly distinguished from the other points, we see $T_4 = 4$. The counting series for rooted trees is denoted by $T(x) = \sum_{P=1}^{\infty} T_P x^P$. We define t_P and $t(x)$ similarly for unrooted trees.

6.7 PARTITIONS

When a positive integer P is expressed as a sum of positive integers $P = \lambda_1 + \lambda_2 + \lambda_3 + \ldots\ldots \lambda_q$, such that $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \ldots\ldots \lambda_q \geq 1$, the q-tuple of called a partition of integer P.

For example, (5), (4 1), (3 2), (3 1 1), (2 2 1), (2 1 1 1), and (1 1 1 1 1) are the seven different partitions of the integer 5.

The integer's λ_i's are called parts of the partitioned number P.

The number of partitions of a given integer P is often obtained with the help of generating function.

The coefficient of x^k in the polynomial

$$(1 + x)(1 + x^2)(1 + x^3) \ldots\ldots (1 + x^P)$$

gives the number of partitions, without repetition, of an integer $k \leq P$.

6.8 GENERATING FUNCTIONS

A generating function $f(x)$ is a power series

$$f(x) = a_0 + a_1 x + a_2 x^2 + \ldots\ldots$$

is some dummy variable x. The coefficient a_k of x^k is the desired number, which depends on a collection of k objects being enumerated.

For example, in the generating function

$$(1 + x)^n = \binom{n}{0} + \binom{n}{1}x + \binom{n}{2}x^2 + \binom{n}{3}x^3 + \ldots\ldots \binom{n}{n}x^n$$

The coefficient of x^k gives the number of distinct combinations of n different objects taken k at a time.

Consider the following generating function :

$$(1 - x)^{-n} = (1 + x + x^2 + x^3 + ...)^n = \sum_{k=0}^{\infty} \binom{n+k-1}{k} x^k. \qquad ...(1)$$

The coefficient of x^k in (1) gives the ways of selecting k objects from n objects with unlimited repetitions.

6.9 COUNTING UNLABELED TREES

The problem of enumeration of unlabeled trees is more involved and requires familiarity with the concepts of generating functions and partitions.

6.10 ROOTED UNLABELED TREES

Let u_n be the number of unlabeled, rooted trees of n vertices and let $u_n(m)$ be the number of those rooted trees of n vertices in which the degree of the root is exactly m. Then

$$u_n = \sum_{m=1}^{n-1} u_n(m).$$

For example, In Fig. 6.4 below, an 11-vertex, rooted tree is composed of four rooted subtrees.

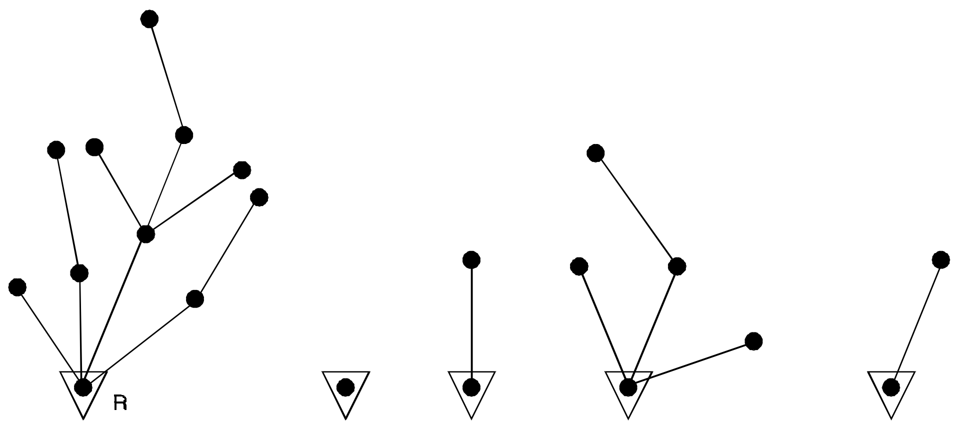


Fig. 6.4. Rooted tree decomposed into rooted subtrees.

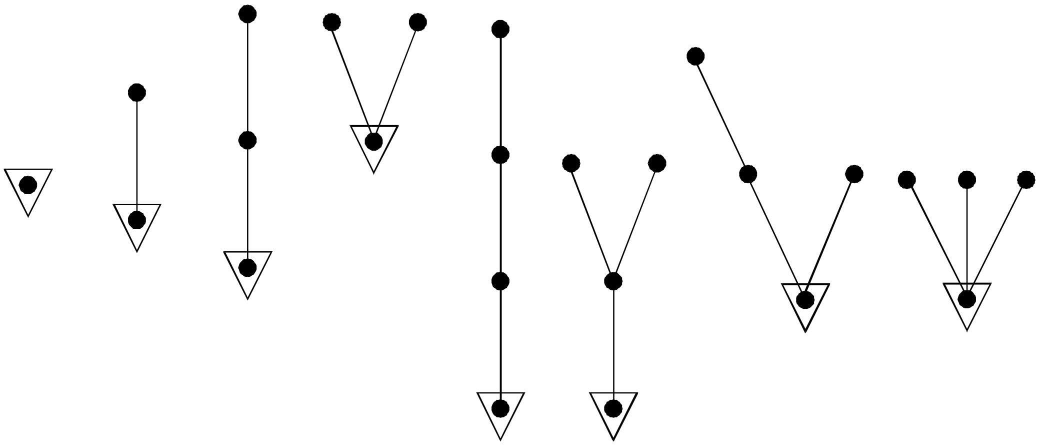


Fig. 6.5. Rooted, ublabeled trees of one, two, three, and four.

6.11 COUNTING SERIES FOR u_n

To circumvent some of these difficulties in computation of u_n, let us find its counting series, *i.e.*, the generating function, $u(x)$, where

$$u(x) = u_1x + u_2x^2 + u_3x^3 + \ldots\ldots.$$

$$= \sum_{n=1}^{\infty} u_n x^n = x \sum_{n=1}^{\infty} u_n x^{n-1}.$$

6.12 FREE UNLABELED TREES

Let $t'(x)$ be the counting series for centroidal trees and $t''(x)$ be the counting series for bicentroidal trees. Then $t(x)$, the counting series for all trees, is the sum of the two. That is $t(x) = t'(x) + t''(x)$.

Thus the number of bicentroidal trees with $n = 2m$ vertices is given by $t_n'' = \binom{u_m + 1}{2} = \frac{u_m(u_m + 1)}{2}$

and
$$t''(x) = \sum_{m=1}^{\infty} \frac{u_m(u_m+1)}{2} x^{2m}$$

$$= \frac{1}{2} \sum_{m=1}^{\infty} u_m x^{2m} + \frac{1}{2} \sum_{m=1}^{\infty} (u_m x^m)^2$$

$$= \frac{1}{2} u(x^2) + \frac{1}{2} \sum_{m=1}^{\infty} (u_m x^m)^2.$$

6.13 CENTROID

In a tree T, at any vertex v of degree d, there are d subtrees with only vertex v in common. The weight of each subtree at v is defined as the number of branches in the subtree. Then the weight of the vertex v is defined as the weight of the heaviest of the subtrees at v. A vertex with the smallest weight in the entire tree T is called a centroid of T.

Every tree has either one centroid or two centroids. If a tree has two centroids, the centroids are adjacent.

In Fig. 6.6 below, a tree with centroid, called a centroidal tree, and a tree with two centroids, called a bicentroidal tree. The centroids are shown enclosed in circles, and the numbers next to the vertices are the weights.

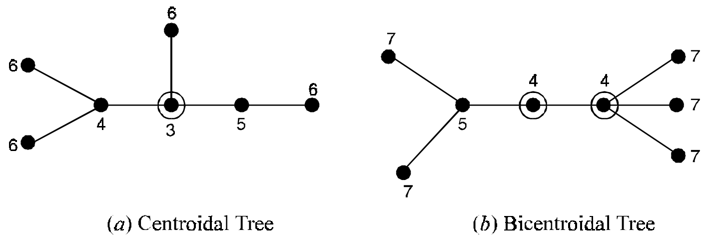


Fig. 6.6. Centroid and bicentroids.

6.14 PERMUTATION

On a finite set A of some objects, a permutation π is a one-to-one mapping from A onto intself.

For example, consider a set $\{a, b, c, d\}$.

A permutation $\pi_1 = \begin{pmatrix} a & b & c & d \\ b & d & c & a \end{pmatrix}$ takes a into b, b into d, c into c, and d into a.

Alternating, we could write $\pi_1(a) = b$, $\pi_1(b) = d$, $\pi_1(c) = c$, $\pi_1(d) = a$.

The number of elements in the object set on which a permutation acts is called the degree of the permutation.

For example, the permutation $\pi_1 = \begin{pmatrix} a & b & c & d \\ b & d & c & a \end{pmatrix}$ is represented diagrammatically by Figure 6.7 below :

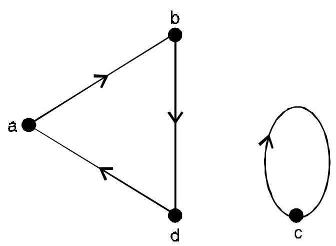


Fig. 6.7. Digraph of a permutation.

Permutation $\begin{pmatrix} a & b & c & d \\ b & d & c & a \end{pmatrix}$ can be written as $(a\ b\ d)\ (c)$.

The number of edges in a permutation cycle is called the length of the cycle in the permutation.

A permutation π of degree k is said to be of type $(\sigma_1, \sigma_2, ... \sigma_k)$ if π has σ_i cycles of length i for $u = 1, 2, ... k$.

For example, permutation is of type (2, 0, 2, 0, 0, 0, 0, 0).

Clearly, $1\sigma_1 + 2\sigma_2 + 3\sigma_3 + ... + k\sigma_k = k$.

6.15 COMPOSITION OF PERMUTATION

Consider the two permutations π_1 and π_2 on an object set {1, 2, 3, 4, 5} : $\pi_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 4 & 5 & 3 \end{pmatrix}$ and $\pi_2 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 4 & 1 & 2 & 5 \end{pmatrix}$.

A composition of these two permutations $\pi_2\pi_1$ is another permutation obtained by first applying π_1 and then applying π_2 on the resultant.

That, is
$$\begin{aligned} \pi_2\pi_1(1) &= \pi_2(2) = 4 \\ \pi_2\pi_1(2) &= \pi_2(1) = 3 \\ \pi_2\pi_1(3) &= \pi_2(4) = 2 \\ \pi_2\pi_1(4) &= \pi_2(5) = 5 \\ \pi_2\pi_1(5) &= \pi_2(3) = 1 \end{aligned}$$

Thus $\pi_2\pi_1 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 2 & 5 & 1 \end{pmatrix}$.

6.16 PERMUTATION GROUP

A collection of m permutations $P = \{\pi_1, \pi_2, ..., \pi_m\}$ acting on a set $A = \{a_1, a_2, ..., a_k\}$ forms a group under composition, if the four postulates of a group, that is, closure, associativity, identity, and inverse are satisfied. Such a group is called a permutation group.

The number of permutations m in a permutation group is called its order, and the number of elements in the object set on which the permutations are acting is called the degree of the permutation group.

For example, the set of four permutations

$\{(a)\ (b)\ (c)\ (d), (a\ c)\ (b\ d), (a\ b\ c\ d), (a\ d\ c\ b)\}$ acting on the object set $\{a, b, c, d\}$ forms a permutaion group.

6.17 CYCLE INDEX OF A PERMUTATION GROUP

For a permutation group P, of order m, if we add the cycle structures of all m permutations in P and divide the sum by m, we get an expression called the cycle index Z(P) of P.

For example, the cycle index of S_3, the full symmetric group of degree three,

$$Z(S_3) = \frac{1}{6}(y_1^3 + 3y_1y_2 + 2y_3).$$

The cycle index of the permutation group is

$$\frac{1}{4}(y_1^4 + y_2^2 + 2y_4)$$

We have six permutations of type (2, 1, 0, 0) on the object set $\{a, b, c, d\}$:

$(a)\ (b)\ (c\ d), (a)\ (c)\ (b\ d), (a)\ (d)\ (b\ c),$
$(b)\ (c)\ (a\ d), (b)\ (d)\ (a\ c), (c)\ (d)\ (a\ b).$

The cycle index of S_4 : $Z(S_4) = \frac{1}{24}(y_1^4 + 6y_1^2y_2 + 8y_1y_3 + 3y_2^2 + 6y_4).$

Permutation type	*Number of such permutations*	*Cycle structures*
(4, 0, 0, 0)	1	y_1^4
(2, 1, 0, 0)	6	$y_1^2y_2$
(1, 0, 1 , 0)	8	y_1y_3
(0, 2, 0, 0)	3	y_2^2
(0, 0, 0, 1)	6	y_4

6.18 CYCLE INDEX OF THE PAIR GROUP

When the n vertices of a group G are subjected to permutation, the $\frac{n(n-1)}{2}$ unordered vertex pair also get permuted.

For example, Let V = $\{a, b, c, d\}$ be the set of vertices of a four-vertex graph. The permutation $\beta = \begin{pmatrix} a & b & c & d \\ d & b & a & c \end{pmatrix}$ on the vertices induces the following permutation on the six unordered vertex pairs :

$$\beta' = \begin{pmatrix} ab & ac & ad & bc & bd & cd \\ db & da & dc & ba & bc & ac \end{pmatrix}.$$

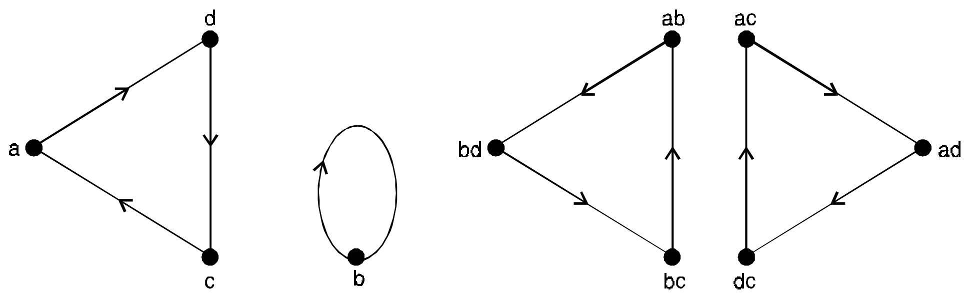


Fig. 6.8. Permutation on vertex set and the induced permutation on vertex-pair set.

6.19 EQUIVALENCE CLASSES OF FUNCTIONS

Consider two sets D and R, with the number of elements | D | and | R | respectively. Let f be a mapping or function which maps each element d from doamin D to a unique image $f(d)$ in range R. Since each of the | D | elements can be mapped into any of | R | elements, the number of different functions from D to R is $|R|^{|D|}$.

Let there be a permutation group P on the elements of set D. Then define two mappings f_1 and f_2 as P-equivalent if there is some permutation π in P such that for every d in D we have

$$f_1(d) = f_2[\pi(d)] \qquad ...(1)$$

The relationship defined by (1) is an equivalence relation can be shown as follows :

(*i*) *Since P is a permutation group, it contains the identity permutation and thus (1) is reflexive.*

(*ii*) *If P contains permutation π, it also contains the inverse permutation π^{-1}. Therefore, the relation is symmetric also.*

(*iii*) *Furthermore, if P contains permutations π_1 and π_2, it must also contain the permutation $\pi_1\pi_2$. This makes P-equivalence a transitive relation.*

The permutation group P on D is the set of all those permutations that can be produced by rotations of the cube. These permutations with their cycle structures are :

(*i*) *One identity permutation. Its cycle structure is y_1^8.*

(*ii*) *Three 180° rotations around lines connecting the centers of opposite faces. Its cycles structure is y_2^4.*

(*iii*) *Six 90° rotations (clockwise and counter clockwise) around lines connecting the centers of opposite faces. The cycle structure is y_4^2.*

(*iv*) *Six 180° rotations around lines connecting the mid-points of opposite edges. The corresponding cycle structure is y_2^4.*

(*v*) *Eight 120° rotations around lines connecting opposite corners in the cube. The cycle structure of the corresponding permutation is $y_1^2y_3^3$.*

The cycle index of this group consisting of these 24 permutations is, therefore,

$$Z(P) = \frac{1}{24}(y_1^8 + 9y_2^4 + 6y_4^2 + 8y_1^2y_3^2).$$

Theorem 6.1. *There are n^{n-2} labeled trees with n vertices ($n \geq 2$).*

Proof. Let the n vertices of a tree T be labeled 1, 2, 3, ..., n. Remove the pendant vertex (and the edge incident on it) having the smallest label, which is, say, a_1.

Suppose that b_1 was the vertex adjacent to a_1.

Among the remaining $n - 1$ vertices.

Let a_2 be the pendant vertex with the smallest label, and b_2 be the vertex adjacent to a_2. Remove the edge (a_2, b_2).

This operation is repeated on the remaining $n - 2$ vertices, and then on $n - 3$ vertices, and so on.

The process is terminated offer $n - 2$ steps, when only two vertices are left.

The tree T defines the sequence $(b_1, b_2, ..., b_{n-2})$ uniquely ...(1)

For example, for the tree in Fig. 6.9(a) below the sequence is (1, 1, 3, 5, 5, 5, 9).

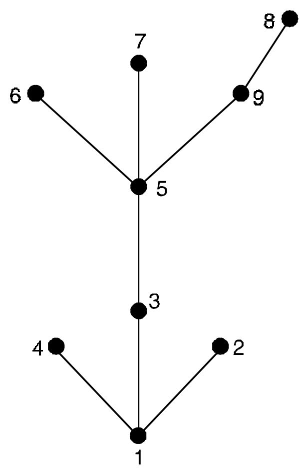


Fig. 6.9.(a) Nine vertex labeled tree, which yields sequence (1, 1, 3, 5, 5, 5, 9).

We note that a vertex i appears in sequence (1) if and only if it is not pendant.

Conversely, given a sequence (1) of $n - 2$ labels, an n-vertex tree can be constructed uniquely, as follows : Determine the first number in the sequence 1, 2, 3, ..., n .... (2) that does not appear in sequence (1).

This number clearly is a_1. And thus the edge (a_1, b_1) is defined. Remove b_1 from sequence (1) and a_1 from (2).

In the remaining sequence of (2) find the first number that does not appear in the remainder of (1). This would be a_2, and thus the edge (a_2, b_2) is defined.

The construction is continued till the sequence (1) has no element left.

Finally, the last two vertices remaining in (2) are joined.

For example, given a sequence (4, 4, 3, 1, 1).

We can construct a seven-vertex tree as follows : (2, 4) is the first edge. The second is (5, 4). Next, (4, 3). Then (3, 1), (6, 1), and finally (7, 1), as shown in Fig. 6.9(*b*) below :

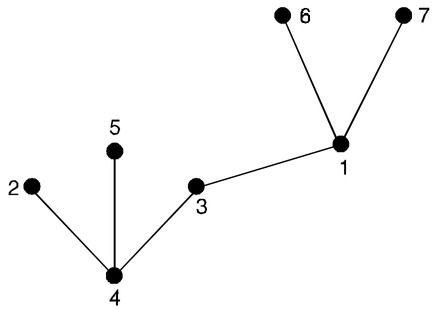


Fig. 6.9.(*b*) Three constructed from sequence (4, 4, 3, 1, 1).

For each of the $n - 2$ elements in sequence (1) we can choose any one of n numbers, thus forming n^{n-2} ...(3)

$(n - 2)$-tuples, each defining a distinct labeled tree of n vertices. And since each tree defines one of these sequences uniquely, there is a one-to-one correspondence between the trees and the n^{n-2} sequences. Hence the theorem.

Theorem 6.2. *The number of different rooted, labeled trees with n vertices is* n^{n-1}.

Theorem 6.2(*a*). ***Pólya's theorem***

The configuration-counting series B(x) is obtained by substituting the figure-counting $A(x^i)$ *for each* y_i *in the cycle index* $Z(P\ ;\ y_1, y_2, \ldots\ldots\ y_k)$ *of the permutation group P.*

That is, $B(X) = Z(P\ ;\ \Sigma a_q x^q, \Sigma a_q x^{2q}, \Sigma a_q x^{3q}, \ldots\ldots, \Sigma a_q x^{kq})$.

Theorem 6.3. *Let A be a permutation group acting on set X with orbits* $\theta_1, \theta_2, \ldots\ldots, \theta_n$*, and W be a function which assigns a weight to each orbit. Furthermore, W is defined on X so that* $w(x) = W(\theta_i)$ *whenever* $x \in \theta_i$*. Then the sum of the weights of the orbits is given by* $|A| \sum_{i=1}^{n} W(\theta_i) = \sum_{\alpha \in A} \sum_{x = \alpha x} W(x)$.

Proof. We have, the order $|A|$ of the group A is the product $|A(x)| . |\theta(x)|$ for any x in X, where $A(x)$ is the stabilizer of x.

Also, since the weight function is constant on the elements in a given orbit, we see that

$$|\theta_i|\, W(\theta_i) = \sum_{x \in \theta_i} W(x), \text{ for each orbit } \theta_i.$$

Combining these facts, we find that

$$|A|\, W(\theta_i) = \sum_{x \in \theta_i} |A(x)|\, W(x)$$

Summing over all orbits, we have

$$|A| \sum_{i=1}^{n} W(\theta_i) = \sum_{i=1}^{n} \sum_{x \in \theta_i} |A(x)|\, W(x)$$

Corollary (BURNSIDE'S LEMMA)

The number N(A) of orbits of the permutation group A is given by $N(A) = \dfrac{1}{|A|} \sum_{\alpha \in A} j_i(\alpha)$.

Proof. Let A be a permutation group of order m and degree d. The cycle index Z(A) is the polynomial in d variables $a_1, a_2, \ldots\ldots a_d$ given by the formula

$$Z(A) = \frac{1}{|A|} \sum_{\alpha \in A} \prod_{k=1}^{d} a_k^{jk(\alpha)}$$

Since, for any permutation α, the numbers $j_k = j_k(\alpha)$ satisfy $1j_1 + 2j_2 + \ldots\ldots d_{jd} = d$ they constitute a partition of the integer d.

The vector notation $(j) = (j_1, j_2, \ldots.., j_d)$ in describing α.

For example, the partition $5 = 3 + 1 + 1$ corresponds to the vector $(j) = (2, 0, 1, 0, 0)$.

Theorem 6.4. *The number of labeled graphs with P points is* $2^{\binom{p}{2}}$.

Corollary. The number of labeled (p, q) graphs is $\binom{\binom{p}{2}}{q}$.

Theorem 6.5. *The number of ways in which a given graph G can be labeled is* $\dfrac{P!}{|\Gamma(G)|}$.

Proof. Let A be a permutation group acting on the set X of objects. For any element x in X, the orbit of x, denoted $\theta(x)$, is the subset of X which consists of all elements y in X such that for some permutation α in A, $\alpha x = y$.

The stabilizer of x, denoted A(x), is the subgraph of A which consists of all the permutations in A which leaves x fixed.

The result follows from an application of the well-known formula $|\theta(x)| \,.\, |A(x)| = |A|$.

Theorem 6.6. ***Pólya's Enumeration theorem***

The configuration counting series is obtained by substituting the figure counting series into the cycle index of the configuration group, C(x, y) = Z(c(x, y)).

Proof. Let α be a permutation in A, and let $\tilde{\alpha}$ be the corresponding permutation in the power group E^A.

Assume first that f is a configuration fixed by $\tilde{\alpha}$ and that ζ is a cycle of length k in the disjoint-cycle decomposition of α.

Then $f(b) = f(\zeta b)$ for every element b in the representation of ζ, so that all elements permuted by ζ must have the same image under f.

Conversely, if the elements of each cycle of the permutation α have the same image under a configuration f, then $\tilde{\alpha}$ fixes f.

Therefore, all configurations fixed by $\tilde{\alpha}$ are obtained by independently selecting an element r in R for each cycle ζ of α and setting $f(b) = r$ for all b permuted by ζ. Then if the weight W(r) is (m, n) where $m = W_1 r$ and $n = W_2 r$ and ζ has length k, the cycle ζ contributes a factor of $\sum_{r \in R} (x^m y^n)^k$ to the sum

$\Sigma_f = \tilde{\alpha} f \; W(f)$.

Therefore, since $\sum_{r \in R} (x^m \; y^n)^k = c(x^k, y^k)$.

We have, for each α in A,

$$\sum_{f = \tilde{\alpha} f} W(f) = \prod_{k=1}^{s} C(x^k, y^k)^{jk(\alpha)} .$$

Summing both sides of this equation over all permutations α in A (or equivalently over all $\tilde{\alpha}$ in E^A) and dividing both sides by $|A| = |E^A|$,

We obtain,

$$\frac{1}{|E^A|} \sum_{\tilde{\alpha} \in E^A} \sum_{f = \tilde{\alpha} f} W(f) = \frac{1}{|A|} \sum_{\alpha \in A} \prod_{k=1}^{S} C(x^k, y^k)^{jk(\alpha)} .$$

The right hand side of this equation is Z(A, C(x, y)).

To see that the left hand side is C(x, y).

Corollary. If A is a permutation group acting on X, then the number of orbits of n-subsets of X induced by A is the coefficient of x^n in Z(A, 1 + x).

Theorem 6.7. *The counting polynomial for graphs with P points is*

$$g_p(x) = Z(S_P^{(2)}, 1 + x),$$

where $$Z(S_P^{(2)}) = \frac{1}{P!} \sum_{(j)} \frac{P!}{\prod_{k=1}^{P} j_k! \, k^{jk}} \prod_{k=1}^{[P/2]} (a_k a_{2k}^{k-1})^{j_{2k}} .$$

$$\prod_{k=0}^{[(P-1)/2]} a_{2k+1}^{kj_{2k+1}} \prod_{k=1}^{[P/2]} a_k^{k\binom{jk}{2}} \prod_{1 \le r < s < P-1} a_{m(r,s)}^{d(r,s) j_r j_s} .$$

Corollary 1. The counting polynomial for rooted graphs with P points is

$$r_P(x) = Z((S_1 + S_{P-1})^{(2)}, 1 + x).$$

When there are atmost two lines joining each pair of points, we need only replace the figure counting series for graphs by $1 + x + x^2$.

Corollary 2. The counting polynomial for multigraphs with at most two lines joining each pair of points is

$$g_P''(x) = Z(S_P^{(2)}, 1 + x + x^2)$$

For arbitrary multigraphs, the figure counting series becomes

$$1 + x + x^2 + x^3 + \ldots\ldots = \frac{1}{1-x}.$$

Corollary 3. The counting polynomial for multigraphs with P points is $m_P(x) = Z\left(S_P^{(2)}, \frac{1}{1-x}\right)$.

Theorem 6.8. *The counting polynomial for digraphs with P points is* $d_P(x) = Z\left(S_P^{(2)}, 1 + x\right)$.

where
$$Z\left(S_P^{(2)}\right) = \frac{1}{P!}\sum_{(j)} \frac{P!}{\prod_{k=1}^{P} j_k!\,k^{j_k}} \prod_{k=1}^{P} a_k^{(k-1)j_k + 2k\binom{j_k}{2}} \cdot \prod_{1 \le r \le S \le P-1} a_{m(r,S)}^{2 j_r j_s d(rS)}.$$

Theorem 6.9. *The number S_P of self-complementary graphs on P points is $S_P = Z(S_P^{(2)}; 0, 2, 0, 2, \ldots..)$.*

Theorem 6.10. *Identity trees are counted by the equations*

$$U(x) = x \exp \sum_{n=1}^{\infty} (-1)^{n+1} \frac{U(x^n)}{n}$$

$$u(x) = U(x) - \frac{1}{2}[U^2(x) + U(x^2)]$$

The number of identity trees through 12 points is given by

$$u(x) = x + x^7 + x^8 + 3x^9 + 6x^{10} + 15x^{11} + 29x^{12} + \ldots\ldots.$$

Theorem 6.11. *The counting series for rooted trees is given by*

$$T(x) = x \prod_{r=1}^{\infty} (1 - x^r)^{-T_r}.$$

Theorem 6.12. *The counting series for rooted trees satisfies the functional equation*

$$T(x) = x \exp \prod_{r=1}^{\infty} \frac{1}{r}(x^r) \qquad \ldots(1)$$

Proof. Let $T^{(n)}(x)$ be the generating function for those rooted trees in which the root has degree n, so that

$$T(x) = \prod_{n=0}^{\infty} T^{(n)}(x) \qquad ...(2)$$

For example, $T^{(0)}(x) = x$ counts the rooted trivial graph, while the planted trees (rooted at an end point) are counted by $T^{(1)}(x) = xT(x)$.

In general a rooted tree with root degree n can be regarded as a configuration whose figures are the n rooted trees obtained on removing the root. Fig. 6.10(a) below, illustrates this for $n = 3$.

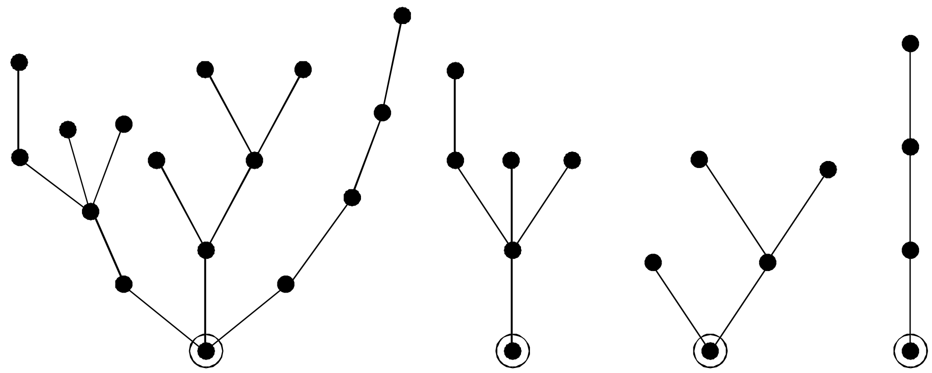


Fig. 6.10(a). A given rooted tree T and its constituent rooted trees.

Since these n rooted trees are mutually interchangeable without altering the isomorphism class of the given rooted tree, the figure counting series is $T(x)$ and the configuration graph is S_n, giving $T^{(n)}(x) = xZ(S_n, T(x))$ ...(3)

The factor x accounts for the removal of the root of the given tree since the weight of a tree is the number of points.

Fortunately, there is a well-known and easily derived identity which may now be invoked (where $Z(S_0)$ is defined as 1)

$$\sum_{n=0}^{\infty} Z(S_n, h(x)) = \exp \sum_{r=1}^{\infty} \frac{1}{r} h(x^r) \qquad ...(4)$$

On combining the last three equations, we obtain (1)

Theorem 6.13. *Homeomorphically irreducible trees are counted by the three equations,*

$$\overline{H}(x) = \frac{x^2}{1+x} \exp \sum_{r=1}^{\infty} \frac{\overline{H}(x^r)}{rx^r} \qquad ...(1)$$

$$H(x) = \frac{1+x}{x} \overline{H}(x) - \frac{1}{2x} [\overline{H}^2(x) - \overline{H}(x)^2] \qquad ...(2)$$

$$h(x) = \mathrm{H}(x) - \frac{1}{x^2}\,[\overline{\mathrm{H}}^2(x) - \overline{\mathrm{H}}(x^2)] \qquad \text{...(3)}$$

The number of homeomorphically irreducible trees through 12 points is found to be :

$$h(x) = x + x^2 + x^4 + x^5 + 2x^6 + 2x^7 + 4x^8 + 5x^9 + 10x^{10} + 14x^{11} + 26x^{12} + \ldots.. \qquad \text{..(4)}$$

Theorem 6.14. ***Power Group Enumeration Theorem***

The number of equivalence classes of functions in R^D determined by the power group B^A is

$$N(B^A) = \frac{1}{|B|}\sum_{\beta \in B}(A\,;m_1(\beta), m_2(\beta), \ldots\ldots, m_1(\beta)) \text{ where } m_k(\beta) = \sum_{s/k} Sj_S(\beta).$$

Theorem 6.15. *The configuration counting series C(x) for 1 – 1 functions from a set of n interchangeable elements into a set with figure counting series C(x) is obtained by substituting C(x) into $Z(A_n - S_n)$:*

$$C(x) = Z(A_n - S_n, C(x)).$$

Theorem 6.16. *For any tree T, let p* and q* be the number of similarity classes of points and lines, respectively, and let S be the number of symmetry lines. Then S = 0 or 1 and*

$$p^* - (q - S) = 1 \qquad \text{...(1)}$$

Proof. Whenever T has one central point or two dissimilar central points, there is no symmetry line, so S = 0.

In this case there is a subtree of T which contains exactly one point from each similarity class of points in T and exactly one line from each class of lines.

Since this subtree has p^* points and q^* lines, we have $p^* - q^* = 1$.

The other posibility is that T has two similar central points and hence S = 1.

In this case there is a subtree which contains exactly one point from each similarity class of points in T and, except for the symmetry line, one line from each class of lines.

Therefore, this subtree has p^* points and $q^* - 1$ lines and so $p^* - (q^* - 1) = 1$.

Thus, in both cases (1) holds.

Theorem 6.17. *The counting series for trees in terms of rooted trees is given by the equation*

$$t(x) = T(x) - \frac{1}{2}[T^2(x) - T(x^2)] \qquad \text{...(1)}$$

Proof. For $i = 1$ to t_n, let p_i^*, q_i^* and S_i be the numbers of similarity classes of points, lines, and symmetry lines for the ith tree with n points.

Since $1 = p_i^* - (q_i^* - S_i)$ for each i, by $p^* - (q^* - S) = 1$, we sum over i to obtain

$$t_n = T_n - \sum_i (q_i^* - S_i) \qquad \text{...(2)}$$

Furthermore $\Sigma(q_i^* - S_i)$ is the number of trees having n points which are rooted at a line, not a symmetry line. Consider a tree T and take any line y of T which is not a symmetry line.

Then T – y may be regarded as two rooted trees which myst be non isomorphic.

Thus each non-symmetry line of a tree corresponds to an unordered pair of different rooted trees.

Counting these pairs of trees is equivalent to counting 1 – 1 functions from a set of two interchangeable elements into the collection of rooted trees.

Therefore, T(x) as the figure counting series to obtain

$$\sum_{n=1}^{\infty}\left[x^n \sum_{i=1}^{t_n}(q_i * - S_i)\right] = Z(A_2 - S_2, T(x)) \qquad ...(3)$$

Since $\quad Z(A_2) = a_1^2$ and $Z(S_2) = \frac{1}{2}(a_1^2 + a_2)$

We have $\quad Z(A_2 - S_2, T(x)) = \frac{1}{2}[T^2(x) - T(x^2)] \qquad ...(4)$

Now the formula in the theorem follows from (2) to (4)

Note :

Using $\qquad T(x) = x \exp \sum_{r=1}^{\infty} \frac{1}{r} T(x^r)$ and

$$t(x) = T(x) - \frac{1}{2}[T^2(x) - T(x^2)],$$

We obtain the explicit numbers of rooted and unrooted trees through P = 12.

$T(x) = x + x^2 + 2x^3 + 4x^4 + 9x^5 + 20x^6 + 48x^7 + 115x^8 + 268x^9 + 719x^{10} + 1842x^{11} + 4766x^{12} + ...$

$t(x) = x + x^2 + x^3 + 2x^4 + 3x^5 + 6x^6 + 11x^7 + 23x^8 + 47x^9 + 106x^{10} + 235x^{11} + 551x^{12} + ...$.

6.20 GROUP DEFINITION

The non-empty set A together with a binary operation, denoted by the just a position $\alpha_1\alpha_2$ for α_1, α_2 in A, constitutes a group whenever the following four axioms are satisfied

Axiom (*i*) (closure) : For all α_1, α_2 in A, $\alpha_1\alpha_2$ is also an element of A.

Axiom (*ii*) (associativity) : For all $\alpha_1, \alpha_2, \alpha_3$ in A, $\alpha_1(\alpha_2\alpha_3) = (\alpha_1\alpha_2)\,\alpha_3$

Axiom (*iii*) (identity) : There is an element i in A such that $\quad i\alpha = \alpha i = \alpha$ for all α in A

Axiom (*iv*) (inversion) : If axiom (*iii*) holds, then for each α in A, there is an element denoted α^{-1} such that $\quad \alpha\alpha^{-1} = \alpha^{-1}\alpha = i$.

6.21 PERMUTATION

A one-one mapping from a finite set onto itself is called a permutation.

6.22 PERMUTATION GROUP

The usual composition of mappings provides a binary operation for permutations on the same set. Whenever a collection of permutations is closed with respect to this composition, Axioms (*ii*), (*iii*) and (*iv*) are automatically satisfied and it is called a permutation group.

Note : If a permutation group A acts on object set X then | A | is the order of this group and | X | is the degree.

6.23 ISOMORPHIC GROUPS

If A and B are permutation groups acting on the sets X and Y then $A \cong B$, means that A and B are isomorphic groups.

Here $A \equiv B$ indicates not only isomorphism but that A and B are identical permutation groups.

If there is one-one map $h : A \leftrightarrow B$ between the permutations such that for all α_1, α_2 in A.

$$h(\alpha_1 \alpha_2) = h(\alpha_1)\, h(\alpha_2) \qquad (i.e., A \cong B)$$

Also, if there is a one-one map $f : X \leftrightarrow Y$ between the objects such that for all x in X and α in A,

$$f(\alpha x) = h(\alpha)\, f(x) \quad (i.e., A \equiv B).$$

6.24 AUTOMORPHISM OF A GROUP

An automorphism of a group G is an isomorphism of G with itself. Thus each automorphism α of G is a permutation of the point set V which preserves adjacency. Of course, α sends any point onto another of the same degree. Obviously any automorphism followed by another is also an automorphism, hence the automorphisms of G form a permutation group $\Gamma(G)$.

6.25 LINE-GROUP

The point-group of G induces another permutation group $\Gamma_1(G)$, called the line-group of G which acts on the liens of G.

6.26 OPERATIONS ON PERMUTATION GROUPS

6.26.1. Sum group

Let A be a permutation group of order $m = |A|$ and degree d acting on the set $X = \{x_1, x_2, ... x_d\}$ and let B be another permutation group of order $n = |B|$ and degree e actng on the set $Y = \{y_1, y_2, ... y_e\}$.

The sum A + B is a permutation group which acts on the disjoint union $X \cup Y$ and whose elements are all the ordered pairs of permutations α in A and β in B, written $\alpha + \beta$. Any object Z of $X \cup Y$ is permuted by $\alpha + \beta$ according to the rule

$$(\alpha + \beta)(Z) = \begin{cases} \alpha z & , \quad z \in X \\ \beta z & , \quad z \in Y \end{cases}.$$

6.26.2. Product Group

The product $A \times B$ of A and B is a permutation group which acts on the set $X \times Y$ and whose permutations are all the ordered pairs written $\alpha \times \beta$ of permutations α in A and β in B. The object (x, y) of $X \times Y$ is permuted by $\alpha \times \beta$ as expected

$$(\alpha \times \beta)(x, y) = (\alpha x, \beta y).$$

6.26.3. Composition Group

The composition A[B] of 'A around B' also acts on $X \times Y$. For each α in A and any sequence $(\beta_1, \beta_2, ... \beta_d)$ of d permutations in B, there is a unique permutation in A[B] written $(\alpha ; \beta_1, \beta_2, ... \beta_d)$ such that for (x_i, y_i) in $X \times Y$.

$$(\alpha ; \beta_1, \beta_2, ..., \beta_d)\,(x_i, y_i) = (\alpha x_i, \beta_i y_j)$$

6.26.4. Power Group

The power group denoted by B^A acts on Y^X, the set of all functions from X into Y. Assume that the power group acts on more than one function. For each pair of permutations α in A and β in B there is a unique permutation, written β^α in B^A. The action of β^α on any function f in Y^X by the following equation which gives the image of each $x \in$ X under the function $\beta^\alpha f : (\beta^\alpha f)(x) = \beta f(\alpha x)$.

6.27 SYMMETRIC GRAPHS

Two points u and v of the graph G are similar if for some automorphism α of G, $\alpha(u) = v$. A fixed point is not similar to any other point. Two lines $x_1 = u_1v_1$ and $x_2 = u_2v_2$ are called similar if there is an automorphism α of G such that $\alpha(\{u_1, v_1\}) = \{u_2, u_2\}$.

A graph is point-symmetric of every pair of points are similar, it is line-symmetric if every pair of lines are similar, and it is symmetric if it is both point-symmetric and line-symmetric.

6.28 HIGHLY SYMMETRIC GRAPHS

A graph G is n-transitive, $n \geq 1$, if it has an n-route and if there is always an automorphism of G sending each n-route onto any other n-route. Obviously a cycle of any length is n-transitive for all n, and a path of length n is n-transitive. We note that not every line-symmetric graph is 1-transitive.

Theorem 6.18. *The line-group and the point-group of a graph G are isomorphism if and only if G has atmost one isolated point and K_2 is not a component of G.*

Proof. Let α' be the permutation in $\Gamma_1(G)$ which is induced by the permutation α in $\Gamma(G)$.

By the definition of multiplication in $\Gamma_1(G)$, we have

$$\alpha'\beta' = (\alpha\beta)' \text{ for all } \alpha, \beta \text{ in } \Gamma(G).$$

Thus the mapping $\alpha \to \alpha'$ is a group homomorphism from $\Gamma(G)$ onto $\Gamma_1(G)$.

Hence $\Gamma(G) \cong \Gamma_1(G)$ if and only if the kernel of this mapping is trivial.

To prove the necessity, assume $\Gamma(G) \cong \Gamma_1(G)$.

Then $\alpha \neq i$ (the identity permutation) implies $\alpha' \neq i$.

If G has distinct isolated points v_1 and v_2, we can define $\alpha \in \Gamma(G)$ by $\alpha(v_1) = v_2$, $\alpha(v_2) = v_1$ and $\alpha(v) = v$ for all $v \neq v_1, v_2$. Then $\alpha \neq i$ but $\alpha' = i$.

If K_2 is a component of G, take the line of K_2 to be $x = v_1v_2$ and define $\alpha \in \Gamma(G)$ exactly as above to obtain $\alpha \neq i$ but $\alpha' = i$.

To prove the sufficiency, assume that G has at most one isolated point and that K_2 is not a component of G.

If $\Gamma(G)$ is trivial, then obviously $\Gamma_1(G)$ fixes every line and hence $\Gamma_1(G)$ is trivial.

Therefore, suppose there exists $\alpha \in \Gamma(G)$ with $\alpha(u) = v \neq u$.

Then the degree of u is equal to the degree of v. Since u and v are not isolated, this degree is not zero.

Case (*i*). u is adjacent to v. Let $x = uv$. Since K_2 is not a component, the degrees of both u and v are greater than one.

Hence there is a line $y \neq x$ which is incident with u and $\alpha'(y)$ is incident with v.

Therefore $\alpha'(y) \neq y$ and so $\alpha' \neq i$.

Case (*ii*). u is not adjacent to v. Let x be any line incident with u. Then $\alpha'(x) \neq x$ and so $\alpha' \neq i$.

Theorem 6.19. *A graph and its complement have the same group* $\Gamma(\overline{G}) = \Gamma(G)$.

Theorem 6.20(*a*). *The group* $\Gamma(G)$ *is* S_P *if and only if* $G = K_P$ *or* $G = \overline{K_P}$.

(***b***) *If G is a cycle of length P, then* $\Gamma(G) = D_P$.

Theorem 2.21. *The three groups* $A + B$, $A \times B$, *and* B^A *are isomorphic.*

Theorem 2.22. *Every finite group F is isomorphic with the group of those automorphisms of D(F) which preserve arc colors.*

Theorem 2.23. *For every finite abstract group F, there exists a graph G such that* $\Gamma(G)$ *and F are isomorphic.*

Theorem 2.24. *If G is a connected graph, then*

$$\Gamma(nG) = S_n[\Gamma(G)].$$

Theorem 2.25. *If* G_1 *is not totally disconnected, then the group of the composition of two graphs* G_1 *and* G_2 *is the composition of their graphs,* $\Gamma(G_1[G_2]) \equiv \Gamma(G_1)\ [\Gamma(G_2)]$, *if and only if the following two conditions hold :*

(*i*) *If there are two points in* G_1 *with the same neighbourhood, then* G_2 *is connected.*

(*ii*) *If there are two points in* G_1 *with the same closed neighbourhood, then* $\overline{G_2}$ *is connected.*

Theorem 6.26. *If* G_1 *and* G_2 *are disjoint, connected, non isomorphic graphs, then* $\Gamma(G_1 \cup G_2) \equiv \Gamma(G_1) + \Gamma(G_2)$.

Corollary (1). The group of the union of two graphs is the sum of their groups

$$\Gamma(G_1 \cup G_2) \equiv \Gamma(G_1) + \Gamma(G_2)$$

if and only if no component of G_1 is isomorphic with a component of G_2.

Corollary (2). The group of the join of two graphs is the sum of their groups

$$\Gamma(G_1 + G_2) \equiv \Gamma(G_1) + \Gamma(G_2)$$

if and only if no component of $\overline{G}_1$ is isomorphic with a component of $\overline{G}_2$.

Theorem 6.27. *The group of the product of two graphs is the product of their groups.*

$$\Gamma(G_1 \times G_2) \equiv \Gamma(G_1) \times \Gamma(G_2)$$

if and only if G_1 and G_2 are relatively prime.

Theorem 6.28. *The group of the corona of two graphs* G_1 *and* G_2 *can be written explicitly in terms of the composition of their groups.*

$$\Gamma(G_1 \bullet G_2) \equiv \Gamma(G_1)\ [E_1 + \Gamma(G_2)].$$

if and only if G_1 *or* $\overline{G}_2$ *has no isolated points.*

Corollary. The group of the corona $G_1 \bullet G_2$ of two graphs is isomorphic to the composition $\Gamma(G_1)\ [\Gamma(G_2)]$ of their groups if and only if G_1 or $\overline{G}_2$ has no isolated points.

Theorem 6.29. *Given any finite, abstract, non trivial group F and an integer* $j(1 \leq j \leq 4)$, *there are infnitely many non homeomorphic graphs G such that G is connected, has no point fixed by every automorphism,* $\Gamma(G) \cong F$, *and G also has the property* P_j, *defined by*

P_1 : $K(G) = n, n \geq 1$

P_2 : $\chi(G) = n, n \geq 2$

P_3 : G is regular of degree n, $n \geq 3$

P_4 : G is spanned by a subgraph homeomorphic to a given graph.

Corollary. Given any finite group F and integers n and m where $n \geq 3$ and $2 \leq m \leq n$, there are an infinite number of graphs G such that $\Gamma(G) \cong F$, $\chi(G) = m$, and G is regular of degree n.

Theorem 6.30. *There exists an n-cage for all $n \geq 3$. For $n = 3$ to 8 there is a unique n-cage. Each of these n-cages is t-unitransitive for some $t = t(n)$, namely, $t(3) = 2$, $t(4) = t(5) = 3$, $t(6) = t(7) = 4$, and $t(8) = 5$.*

Theorem 6.31. *Whenever $P \geq 20$ is divisible by 4, there exists a regular graph G with P points which is line-symmetric but not point-symmetric.*

Theorem 6.32. *If G is connected, n-transitive, is not cycle, has no end points and has girth g, then $n \leq 1 + \dfrac{g}{2}$.*

Theorem 6.33. *Let G be a connected graph with no end points. If W is an n-route such that there is an automorphism of G from W onto each of its successors, then G is n-transitive.*

Theorem 6.34. *Every line-symmetric graph with no isolated points is point-symmetric or bipartite.*

Proof. Consider a line-symmetric graph G with no isolated points, having q lines.

Then for any line x, there are at least q automorphisms $\alpha_1, \alpha_2, ... \alpha_q$ of G which map x onto the lines of G.

Let $x = v_1 v_2$, $V_1 = \{\alpha_1(v_1), ... \alpha_q(v_1)\}$, and $V_2 = \{\alpha_1(v_2), ... \alpha_q(v_2)\}$.

Since G has no isolated points, the union of V_1 and V_2 is V.

There are two possibilities : V_1 and V_2 are disjoint or they are not.

Case (*i*). If V_1 and V_2 are disjoint then G is bipartite.

Consider any two points u_1 and W_1 in V_1. If they are adjacent, then there is a line y joining them. Hence for some automorphism α_i, we have $\alpha_i(x) = y$.

This implies that one of these two points is in V_1 and the other is in V_2, a contradiction.

Hence V_1 and V_2 constitute a partition of V such that no line joins two points in the same subset.

By definition, G is bipartite.

Case (*ii*). If V_1 and V_2 are not disjoint, then G is point-symmetric.

Let u and w be any two points of G. We wish to show that u and w are similar.

If u and w are both in the same set, say V_1 then there exists automorphism α with $\alpha(v_1) = u$ and β with $\beta(v_1) = w$.

Thus $\beta\alpha^{-1}(u) = w$ so that any two points u and w in the same subset are similar.

If u is in V_1 and w is in V_2, let v be a point in both V_1 and V_2.

Since v is similar with u and with w, u and w are similar to each other.

Corollary (1) : If G is line-symmetric and the degree of every line is (d_1, d_2) with $d_1 \neq d_2$ then G is bipartite.

Corollary (2) : If a graph G with no isolated points is line-symmetric, has an odd number of points, and the degree of every line is (d_1, d_2) with $d_1 = d_2$, then G is point-symmetric.

Corollary (3) : If G is line-symmetric, has an even number of points, and is regular of degree $d \geq \frac{P}{2}$ then G is point-symmetric.

Problem Set 6.1

1. Prove that the number of different rooted labeled trees with n vertices is n^{n-1}.
2. Prove that the number of simple labeled graphs of n vertices is $2^{n(n-1)/2}$.
3. Prove that there are n^{n-2} labeled trees with n vertices $(n \geq 2)$.
4. Prove that a vertex v appears in sequence $(b_1, b_2, \ldots.. b_{n-2})$ m times if and only if degree of $v = m - 1$.
5. Show that the cycle index of a group consisting of the identity permutation only is y_1^k, k being the number of elements in the object set.
6. Prove that the number of labeled graphs with P points is $2^{\binom{P}{2}}$.
7. Prove that the number of ways in which a given graph G can be labeled is $\frac{P!}{|\Gamma(G)|}$.
8. Prove that the number N(A) of orbits of the permutation group A is given by N(A) = $\frac{1}{|A|} \sum_{\alpha \in A} j_1(\alpha)$.
9. Prove that the counting polynomial for rooted graphs with P points is $r_P(x) = Z((S_1 + S_{P-1})^{(2)}, 1 + x)$.
10. Prove that the counting series for rooted trees is given by $T(x) = x \prod_{r=1}^{\infty} (1 - x^r)^{-Tr}$.
11. Show that the cycle index of the induced pair group R_3 is the same as that of S_3, that is
$$Z(R_3) = \frac{1}{6}(y_1^3 + 3y_1y_2 + 2y_3).$$
12. Show that the order of D_n, the group of symmetries of a regular n-side polygon is $2n$. Find the cycle index of D_n.
13. Find the automorphism group Ω(G) of a graph G if G is
   (*i*) a complete graph of n vertices and
   (*ii*) a circuit with n vertices.
   Find a graph with minimum number of vertices $n > 1$ in which Ω(G) consists of only the identity permutation.

14. Show that the cycle index of S_5 the full symmetric group of degree five is

$$Z(S_5) = \frac{1}{5!}(y_1^5 + 10y_1^3y_2 + 20y_1^2y_3 + 15y_1y_2^2 + 30y_1y_4 + 20y_2y_3 + 24y_5).$$

15. List all partition of 5 and use them to find u_6, the number of unlabeled trees of six vertices.

16. Prove that the number of equivalence classes of functions in R^D determined by the power group B^A is $N(B^A) = \frac{1}{|B|}\sum_{\beta \in S} Z(A\,;\, m_1(\beta), m_2(\beta), \ldots\ldots\, m_d(\beta))$ where $m_k(\beta) = \sum_{S/k} S_{js}(\beta)$.

17. Prove that the number S_P of self-complementary graphs on P points is $S_P = Z(S_P^{(2)}\,;\, 0, 2, 0, 2, \ldots\ldots)$.

18. Find the number of trees with P points which are

(*i*) planted and labeled

(*ii*) rooted and labeled.

19. Find counting series for unicyclic graphs.

20. In how many ways can the graphs

(*i*) $\overline{K}_3 + K_2$ (*ii*) $K_3 \times K_2$ (*iii*) $K_{1,\,2}\,[K_2]$ be labeled ?

21. Prove that the number of partitions of n into at most m parts is the coefficient of x^n in $Z\left(S_m, \frac{1}{1-x}\right)$.

22. Define the numbers $R_n^{(i)}$ by the equation $R_n^{(i)} = R_{n-1}^{(i)} + T_{n+1-i}$ then prove that the number of rooted trees can be found using

$$^nT_{n+1} = \sum_{i=1}^{n} i^i T_i\, R_n^{(i)}$$

23. Show that the cycle index of the unordered pair groups R_5 (on the set of 10 unordered pair induced by S_5) is

$$Z(S_5) = \frac{1}{5!}(y_1^{10} + 10y_1^4y_2^3 + 20y_1y_3^3 + 15y_1^2y_2^4 + 30y_2y_4^2 + 20y_2y_3y_6 + 24y_5^2).$$

24. Find the number of different ways of painting the four faces of a pyramid with two colors.

25. Prove that a digraph in which the in-degree as well as the out-degree of every vertex is one can be decomposed into one or more vertex-disjoint directed circuits.

26. Find the counting series for the structural isomers of saturated alcohols $C_nH_{2n+1}OH$.

27. Find the counting series for unlabeled, simple, connected graphs with exactly one circuit.

28. Prove that the number of rooted trees satisfies the inequality $T_{n+1} \le \sum_{i=1}^{n} T_i\, T_{n-i+1}$. It shows that

$$T_n \le \frac{1}{n}\binom{2n-2}{n-1}.$$

29. Let $g(x, y) = \sum_{p=1}^{\infty} g_P(x) y^p$ be the generating function for graphs and let $c(x, y)$ be that for connected graphs then show that

$$g(x, y) = exp \sum_{r=1}^{\infty} \frac{1}{r} c(x^r, y^r) .$$

30. Prove that if a set of permutation P on an object set S forms a group, the set R of all permutations induced by P on set S × S along forms a group.

31. Prove that a subset A of a finite group forms a subgroup if the subset satisfies the closure postulate.

32. Find the different ways of painting the six vertices of an octahedron with three colors.

33. Determine the number S_p of self complementary graphs for P = 8 and 9, both by formula $S_P = Z(S_P^{(2)}; 0, 2, 0, 2, ...)$ and by constructing them.

34. Prove that the configuration counting series is obtained by substituting the figure counting series into the cycle index of the configuration group

$$C(x, y) = Z(A, C(x, y)).$$

35. Prove that the smallest non trivial graph having only the identity endomorphism has 8 points.

36. Let G be a triply connected planar (p, q) graph whose group has order S then show that $\frac{4q}{s}$ is an integer and $S = 4q$ if and only if G is one of the five platonic graphs.

37. Find the groups of the following graphs

(*i*) $\overline{3K_2}$ (*ii*) $\overline{K_2} + C_4$ (*iii*) $K_{m, n}$ (*iv*) $K_{1, 2}[K_2]$ (*v*) $K_4 \cup C_4$.

38. Prove that every symmetric, connected graph of odd degree is 1-transitive.

39. Prove that every, symmetric, connected, cubic graph is *n*-transitive for some *m*.

40. Prove or disprove the following eight statements. If two graphs are point symmetric (line-symmetric) then so are their join, product, composition and corona.

41. Prove that the only connected graph with group isomorphic to S_n, $n \ge 3$

(*i*) with *n* points is K_n

(*ii*) with $n + 1$ points is $K_{1, n}$

(*iii*) with $n + 2$ points is $K_1 + \overline{K_{1, n}}$.

42. Let C(m) be the smallest number of points in a graph whose group is isomorphic to C_m. Then prove that the values of C(m) for $m = n^r$ and n prime are

(i) $C(2) = 2$ and $C(2^r) = 2^r + 6$ when $r > 1$.

(ii) $C(n^r) = n^r + 2n$ for $n = 3, 5$

(iii) $C(n^r) = n^r + n$ for $n \geq 7$.

43. Let G be connected with P > 3. Then show that L(G) is prime if and only if G is not $K_{m,\, n}$ for $m, n \geq 2$.

44. If a connected graph G has a point which is not in a cycle of length four then prove that G is prime.

45. If G is point-symmetric then prove that, if Γ(G) is abelian, it is a group of the form $S_2 + S_2 + \ldots\ldots + S_2$.

46. Find the necessary and sufficient conditions for the point-group and line-group of a graph to be indentical.

47. Let A and B be two permutation groups acting on the sets $X = \{x_1, x_2, \ldots\ldots x_d\}$ and Y respectively. The exponentiation group, denoted $[B]^A$, acts on the functions Y^X. For each permutation α in A and each sequence of permutation $\beta_1, \beta_2, \ldots\ldots \beta_d$ in B there is a unique permutation $[\alpha\, ; \beta_1, \beta_2, \ldots\ldots \beta_d]$ in $[B]^A$ such that for x_i in X and f in Y^X

$$[\alpha\, ; \beta_1, \beta_2, \ldots.. \beta_d]\, f(x_i) = \beta_i\ f(\alpha x_i)$$

Then prove that the group of the cube Q_n is $[S_2]^{S_n}$ and the line-group of $K_{n,\, n}$ is $[S_n]^{S_2}$.

CHAPTER

7 Coverings, Partitions and Factorization

7.1 COVERINGS

A point and a line are said to **cover** each other if they are **incident**. A set of points which covers all the lines of a graph G is called a **point cover** for G, while a set of lines which covers all the points is a **line cover**.

7.1.1. Point covering number and line covering number

The smallest number of points in any point cover for G is called its point **covering number** and is denoted by α_0 (G) or α_0. Similarly α_1 (G) or α_1 is the smallest number of lines in any line cover of G and is called its **line covering number**.

For example. α_0 (K_P) = P – 1 and α (K_P) = [(P + 1)/2].

A point cover or line cover is called minimum if it contains α_0 (respectively α_1) dements.

We observe that a point cover may be minimum without being minimum, such a set of points is given by the 6 non cut points in Fig. 7.1(*a*) below. The same holds for line covers, the 6 lines incident with the cut point serve.

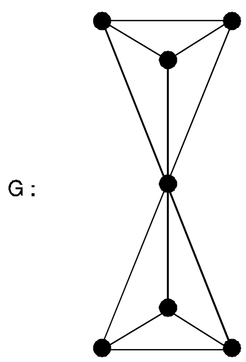


Fig. 7.1.(*a*) The graph k_4.

7.2 INDEPENDENCE

A set of points in G is independent if no two of them are adjacent.

7.2.1. Point independence number

The largest number of points in such a set is called the point is independence number of G and is denoted by β_0 (G) or β_0.

7.2.2. Line independence number

An independent set of lines of G has no two of its lines adjacent and the maximum cardinality of such a set is the line independence number β_1 (G) or β_1.

For the complete graph, β_0 (K_P) = 1 and β_1 (K_P) = [P/2].

From the above graph, β_0 (G) = 2 and β_1(G) = 3.

7.3 VERTEX COVERING

A subset W of V is called a vertex covering or a vertex cover of G if every edge in G is incident on at least one vertex in W.

7.3.1. Trivial vertex covering

A vertex cover of a graph is a subgraph of the graph, V it self is a vetex covering of G. This is known as the trivial vertex covering.

7.3.2. Minimal vertex covering

A vertex covering W of G is called a minimal vertex covering if no proper subset of W is a vertex covering of G.

For example. In the graph shown in Fig. 7.1(*b*) below, the set W = {v_2, v_4, v_6} is a vertex covering.

We check that {v_1, v_2}, {v_1, v_3}, {v_2, v_3}, {v_1}, {v_2}, {v_3} are not vertex coverings of the graph. Thus, no proper subset of W is a vertex covering. Hence W is a minimal vertex covering.

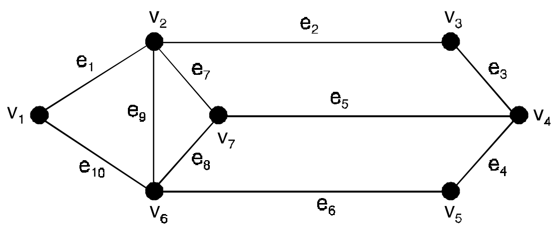


Fig. 7.1.(*b*)

7.4 EDGE COVERING

A non empty subset S of E is called an edge covering or an edge cover of G if every non isolated vertex in G is inciedent with at least one edge in S.

7.4.1. Trivial edge covering

An edge cover of a graph is a subgraph of the graph, E itself is an edge covering of G. This is known as the trivial edge covering.

7.4.2. Minimal edge covering

An edge covering S of G is called a minimal edge covering if no proper subset of S is an edge covering of G.

For example. In Figure 7.1(*b*), the set S = {e_1, e_3, e_6, e_8} is an edge covering.

7.5 CRITICAL POINTS AND CRITICAL LINES

If H is a subgraph of G, then α_0 (H) ≤ α_0 (G). In particular this inequality holds when H = G – *v* or H = G – *x* for any point *v* or line *x*.

If α_0 (G – *v*) < α_0 (G) then *v* is called a critical point, if α_0 (G – *x*) < α_0 (G) then *x* is a critical line of G.

If *v* and *x* are critical, it follows that

$$\alpha_0 (G - v) = \alpha_0 (G - x) = \alpha_0 - 1.$$

7.6 LINE-CORE AND POINT-CORE

The line-core C_1 (G) of a graph G is the subgraph of G induced by the union of all independent sets Y of lines (if any) such that |Y| = α_0 (G).

For example. Consider an odd cycle C_P. Here we find that α_0 (C_P) = (P + 1)/2 but that β_1 (C_P) = (P – 1)/2 so C_P has no line-core.

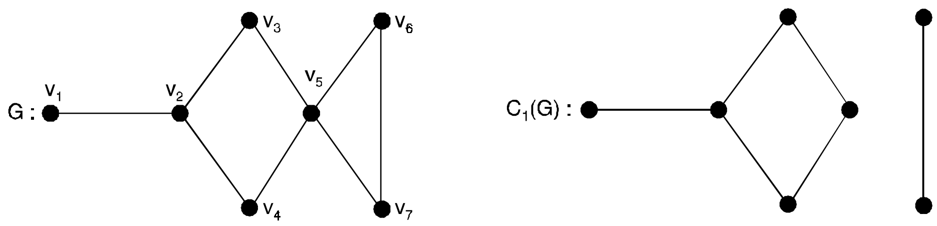


Fig. 7.2. A graph and its line-core.

A minimum point cover M for a graph G with point set V is said to be external if for each subset M′ of M, |M′| ≤ |U (M′)|, where U (M′) is the set of all points of V – W which are adjacent to a point of M′.

Observations

(*i*) A covering exists for a graph if and only if the graph has no isolated vertex.

(*ii*) A covering of an *n*-vertex graph will have at least [*n*/2] adges. $(\lceil x \rceil$ denotes the smallest integer not less than $x)$

(*iii*) Every pendent edge in a graph is include in every covering of the graph.

(*iv*) Every covering contains a minimal covering.

(*v*) If we denote the remaining edges of a graph by (G – *g*), the set of edges *g* is a covering if and only if, for every vertex *v*, the degree of vertex in (G – *g*) ≤ (degree of vertex *v* in G) – 1.

(*vi*) No minimal covering can contain a circuit, for we can always remove an edge from a circuit without leaving any of the vertices in the circuit uncovered. Therefore, a minimal covering of an *n*-vertex graph can contain no more than *n* – 1 edges.

(*vi*) A graph, in general, has many minimal coverings, and they may be of different sizes (*i.e*,. consisting of different numbers of edges). The number of edges in a minimal covering of the smallest size is called the covering number of the graph.

7.7 PARTITIONS

The degrees d_1, ... d_P of the points of a graph form a sequence of non-negative integers, whose sum is of course $2q$. In number theory, to define a partition of a positive integer n as a list or unordered sequence of positive integers whose sum is n.

For example, 4 has five partitions

$$4,\ 3 + 1,\ 2 + 2,\ 2 + 1 + 1,\ 1 + 1 + 1 + 1.$$

The degrees of a graph with no isolated points determine such a partition of $2q$.

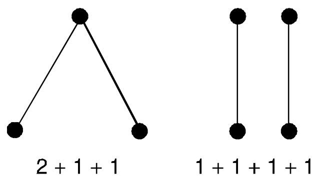


Fig. 7.3. The graphical partitions of 4.

The partition of a graph is the partition of $2q$ as the sum of the degrees of the points $2q = \Sigma d_i$.

A partition Σd_i of n into P parts is graphical if there is a graph G whose points have degree d_i. If such a partition is graphical then certainly every $d_i \le P - 1$ and n is even.

7.8 1-FACTORIZATION

A factor of a graph G is a spanning subgraph of G which is not totally disconnected. We say that G is the sum of factors G_i if it is their line-disjoint union and such a union is called a factorization of G.

If G is the sum of n-factors their union is called an n-factorization and G it self is n-factorable.

When G has a 1-factor, say G_1, it is clear that P is even and the lines of G_1 are point disjoint. In particular, K_{2n+1} cannot have a 1-factor but K_{2n} certainly can.

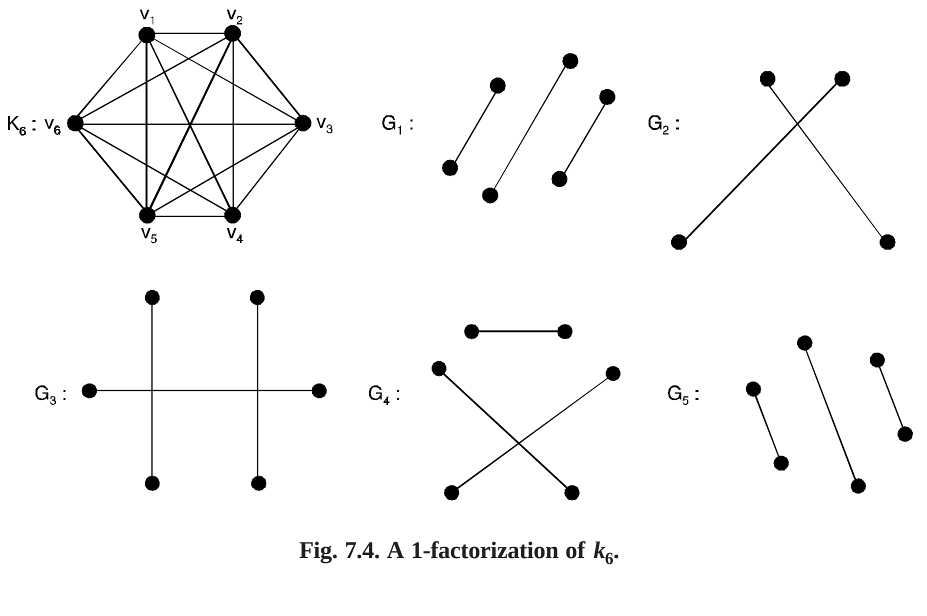


Fig. 7.4. A 1-factorization of k_6.

7.9 2-FACTORIZATION

If a graph is 2-factorable then each factor must be a union of disjoint cycles. If a 2-factor is connected, it is a spanning cycle. We saw that a complete graph is 1-factorable if and only if it has even number of points. Since a 2-factorable graph must have all points even, the complete graphs K_{2n} are not 2-factorable. The odd complete graphs are 2-factorable and infact a stronger statement can be made.

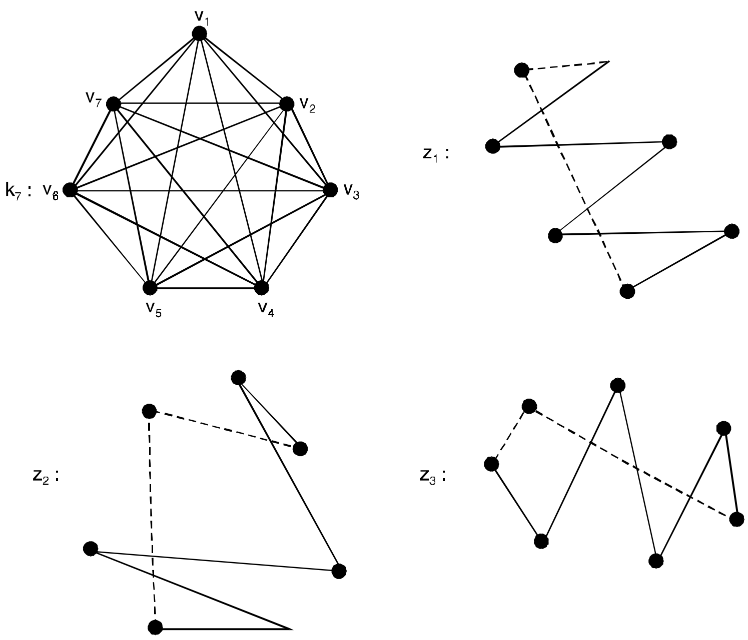


Fig. 7.5. A 2-factorization of k_7.

7.10 ARBORICITY

Any graph G can be expressed as a sum of spanning forests, simply by letting each factor contain only one of the q lines of G. A natural problem is to determine the minimum number of line-disjoint spanning forests into which G can be decomposed. This number is called the arboricity of G and is denoted by r (G).

For example. $r(K_4) = 2$ and $r(K_5) = 3$, minimal decompositions of those graphs into spanning forests are shown in Figure 7.6 below.

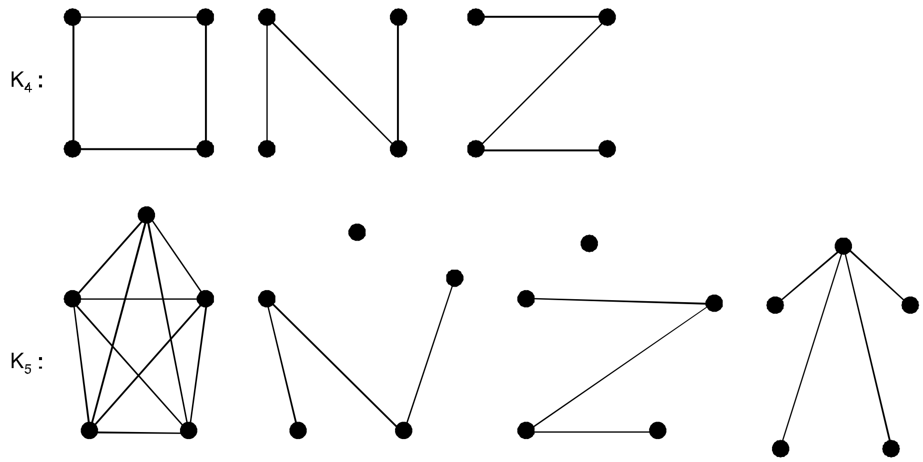


Fig. 7.6. Minimal decompositions into spanning forests.

Theorem 7.1. *For any nontrivial connected graph G,*

$$\alpha_0 + \beta_0 = P = \alpha_1 + \beta_1$$

Proof. Let M_0 be any maximum independent set of points, so that $|M_0| = \beta_0$.

Since no line joins two points of M_0, the remaining set of $P - \beta_0$ points constitutes a point cover for G so that $\alpha_0 \le P - \beta_0$.

On the other hand, if N_0 is a minimum point cover for G, so the set $V - N_0$ is independent.

Hence, $\beta_0 \ge P - \alpha_0$, proving the first equation.

To obtain the second equality, we begin with an independent set M_1 of β_1 lines.

A line cover Y is then produced by taking the union of M_1 and a set of lines one incident line for each point of G not covered by any line in M_1.

Since $|M_1| + |Y| \le P$ and $|Y| \ge \alpha_1$. It follows that $\alpha_1 + \beta_1 \le P$.

In order to show the inequality in the other direction, let us consider a minimum line cover N_1 of G.

Clearly, N_1 cannot contain a line both of whose endpoints are incident with lines also in N_1.

This implies that N_1 is the sum of stars of G (considered as sets of lines).

If one line is selected from each of these stars, we obtain an independent set W of lines.

Now, $|N_1| + |W| = P$ and $|W| \le \beta_1$.

Thus, $\propto_1 + \beta_1 \ge P$, completing the proof of the theorem.

Corollary If P is an hereditary property of G, then

$$\alpha_0 (P) + \beta_0 (P) = P.$$

Theorem 7.2. *A graph G and its line-core C_1 (G) are equal if and only if G is bipartite and not reducible.*

Theorem 7.3. *The following are equivalent for any graph G :*

(*i*) *G has a line-core*

(*ii*) *G has an external minimum point cover.*

(*iii*) *Every minimum point cover for G is external.*

Theorem 7.4. *Any two adjacent critical lines of a graph lie on an odd cycle.*

Corollary (1) Every connected line-critical graph is a block in which any two adjacent lines lie on an odd cycle.

Corollary (2) Any two critical lines of a bipartite graph are independent.

Theorem 7.5. *A point v is critical in a graph G if and only if some minimum point cover contains v.*

Proof. If M is a minimum point cover for G which contains v, then M – {v} covers G – v.

Hence $\alpha_0 (G - v) \leq |M - \{v\}| = |M| - 1 = \alpha_0 (G) - 1$

So that v is critical in G.

Let v be a critical point of G and consider a minimum point cover M′ for G – v.

The set $M' \cup \{v\}$ is a point cover for G, and since it contains one more element than M′, it is minimum.

Theorem 7.6. *For any graph G,*

$$\alpha_{00} \leq \alpha'_{00} \text{ and } \alpha_{11} = \alpha'_{11}$$

Theorem 7.7. *If Y is a line cover of G such that there is no Y-reducing walk, then Y is a minimum line cover.*

Theorem 7.8. *If G is bipartite then the number of lines in a maximum matching equals the point covering number, that is,* $\beta_1 = \alpha_0$.

Theorem 7.9. *Every unaugmentable matching is maximum.*

Proof. Let M be unaugmentable and choose a maximum matching M′ for which |M – M′|, the number of lines which are in M but not in M′ is minimum.

If this number is zero then M = M′.

Otherwise, construct a trail W of maximum length whose lines alternate in M – M′ and M′.

Since M′ is unaugmentable, trail W cannot begin and end with lines of M – M′ and has equally many lines in M – M′ and in M′.

Now we form a maximum matching N from M′ by replacing those lines of W which are in M′ by the lines of W in M– M′.

Then |M – N| < |M – M′|, contradicting the choice of M′ and completing the proof.

Theorem 7.10. *A partition* $2q = \Sigma_1^P d_i$ *belongs to a tree if and only if each* d_i *is positive and* $q = P - 1$.

Theorem 7.11. *Let* $\Pi = (d_1, d_2, \ldots\ldots, d_P)$ *be a partition of 2q into P > 1 parts,* $d_1 \geq d_2 \geq \ldots\ldots \geq d_P$.

Then π *is graphical if and only if for each integer r,* $1 \geq r \geq P - 1$, $\sum_{i=1}^{r} d_i \geq r(r-1) + \sum_{i=r+1}^{P} \min\{r, d_i\}$. ...... (*i*)

Proof. Necessity part :

Given that π is a partition of $2q$ belonging to a graph G, teh sum of the r largest degrees can be considered in two parts, the first being the contribution to this sum of lines joining the corresponding r points with each other, and the second obtained from lines joining one of these r points with one of the remaining P – r points. These two parts are respectively atmost $r(r-1)$ and $\sum_{i=r+1}^{P} \min\{r, d_i\}$.

Sufficiency part :

The proof of the sufficiency is by induction on P. Clearly the result hold for sequences of two parts. Assume that it holds for sequences of P parts, and let $d_1, d_2, \ldots\ldots, d_{P+1}$ be a sequence satisfying the hypotheses of the theorem.

Let m and n be the smallest and largest integers

such that $d_{m+1} = \ldots\ldots = d_{d_1+1} = \ldots\ldots = d_n$.

Form a new sequence of P terms by letting

$$e_i = \begin{cases} d_{i+1} - 1 & \text{for } i = 1 \text{ to } m-1 \text{ and } n-1-(d_1-m) \text{ to } n-1 \\ d_{i+1} & \text{otherwise.} \end{cases}$$

If the hypothesis of the theorem hold for the new sequence $e_1, \ldots\ldots, e_p$, then by the induction hypothesis there will be a graph with the numbers e_i as degrees. A graph having the given degreee sequence d_i will be formed by adding a new point of degree d_1 adjacent to points of degrees corresponding to those terms e_i which were obtained by subtracting 1 from terms d_{i+1} as above.

Clearly $P > e_1 \geq e_2 \geq \ldots\ldots \geq e_p$. Suppose that condition (1) does not hold and let h be the least value of r for which it does not. Then

$$\sum_{i=1}^{h} e_i > h(h-1) + \sum_{i=h+1}^{p+1} \min\{h, e_i\} \qquad \ldots(2)$$

But the following inequalities do hold :

$$\sum_{i=1}^{h+1} d_i \leq h(h+1) + \sum_{i=h+2}^{p+1} \min\{h+1, d_i\} \qquad \ldots(3)$$

$$\sum_{i=1}^{h-1} e_i \leq (h-1)(h-2) + \sum_{i=h}^{p} \min\{h-1, e_i\} \qquad \ldots(4)$$

$$\sum_{i=1}^{h-2} e_i \leq (h-2)(h-3) + \sum_{i=h-1}^{p} \min\{h-2, e_i\} \qquad \ldots(5)$$

Let S denote the number of values of $i \leq h$ for which

$$e_i = d_{i+1} - 1$$

Then (3) – (5) when combined with (2) yield

$$d_1 + S < 2h + \sum_{i=h+1}^{p} \left(\min\{h+1, d_{i+1}\} - \min\{h, e_i\}\right) \qquad ...(6)$$

$$e_h > 2(h-1) - \min\{h-1, e_h\} + \sum_{i=h+1}^{p} \left(\min\{h, e_i\} - \min\{h-1, e_i\}\right) \qquad ...(7)$$

$$e_{h-1} + e_h > 4h - 6 - \min\{h-2, e_{h-1}\} - \min\{h-2, e_h\}$$
$$+ \sum_{i=h+1}^{p} \left(\min\{h, e_i\} - \min\{h-2, e_i\}\right) \qquad ...(8)$$

We note that $e_h + h$ since otherwise inequality (7) gives a contradiction.

Let a, b and c denote the number of values of $i > h$ for which $e_i > h$, $e_i = h$ and $e_i < h$ respectively. Furthermore, let a', b' and c' denote the numbers of these for which $e_i = d_{i+1} - 1$. Then

$$d_1 = s + a' + b' + c' \qquad ...(9)$$

The inequalitites (6) – (8) now become

$$d_1 + s < 2h + a + b' + c' \qquad ...(10)$$

$$e_h \geq \mathrm{h} + a + b \qquad ...(11)$$

$$e_{h-1} + e_h \geq 2h - 1 + \sum_{i=h+1}^{p} \left(\min\{h, e_i\} - \min\{h-2, e_i\}\right) \qquad ...(12)$$

There are now several cases to consider.

Case (*i*) $c' = 0$. Since $d_1 \geq e_h$. We have from (11)

$$h + a + b \leq d_1.$$

But a combination of (9) and (10) gives

$$2d_1 < 2h + a + a' + 2b'$$

which is a contradiction.

Case (*ii*) $c' > 0$ and $d_{h+1} > h$. This means that

$$d_{i+1} = e_i + 1 \text{ whenever } d_{i+1} > h.$$

Therefore since $d_{h+1} > h$, $s = h$ and $a = a'$. But the inequalities (10) and (9) imply that

$$d_1 + h < 2h + a' + b' + c' = d_1 + h, \text{ a contradiction.}$$

Case (*iii*) $c' > 1$ and $d_{h+1} = h$. Under these circumstances

$e_h = h$ and $a = b = 0$, So $d_1 = s + c'$. Furthermore,

Since $e_h = d_{h+1}$, $e_i = h - 1$ for at least c' values of $i > h$.

Hence inequality (12) implies

$$e_{h-1} \geq h - 1 + c' > h.$$

So that $e_{h-1} = d_h - 1$.

Therefore $s = h - 1$ and $d_1 = h - 1 + c' \leq e_{h-1} < d_h$ a contradiction.

Case (iv) $c' = 1$ and $d_{h+1} = h$. Again $e_h = h$, $a = b = 0$, and $d_1 = s + c'$.

Since $s \le h - 1$, $d_1 = h$. But this implies $s = 0$ and $d_1 = 1$, so all $d_i = 1$.

Thus (1) is obviously setisfied, which is a contradiction.

Since $e_h \ge h$ and $d_{h+1} \ge e_h$, we see that d_{h+1} connot be less that h.

Thus all possible cases have been considered and the proof is complete.

Theorem 7.12. *A partition $\pi = (d_1, d_2, ..., d_p)$ of an even number into P parts with $P - 1 \ge d_1 \ge d_2 \ge ...... \ge d_p$ is graphical if and only if the modified partition.*

$$\pi' = (d_2 - 1, d_3 - 1, ..., d_{d_1+1} - 1, d_{d_1+2} ......, d_p)$$

is graphical.

Proof. If π' is graphical, then so is π, since from a graph with partition π' one can construct a graph with partition π by adding a new point adjacent to points of degrees $d_2 - 1$, $d_3 - 1$, ..., $d_{d_1+1} - 1$.

Now let G be a graph with partition π. If a point of degree d_1 is adjacent to points of degree d_i for $i = 2$ to $d_1 + 1$, then the removal of this point results in a graph with partition π'.

Therefore we will show that from G one can get a graph with such a point. Suppose that G has no such point. We assume that in G, Point v_i has degree d_i, with v_1 being a point of degree d_1 for which the sum of the degrees of the adjacent points is maximum. Then there are points v_i and v_j with $d_i > d_j$ such that v_1v_j is a line but v_1v_i is not.

Therefore some point v_k is adjacent to v_i but not to v_j. Removal of the lines v_1v_j and v_kv_i and addition of v_1v_i and v_kv_j results in another graph with partition π in which the sum of the degrees of the points adjacent to v_1 is greater than before. Repeating this process results in a graph in which v_1 has the desired property.

Corollary. *(Algorithm)*

A given partition $\pi = (d_1, d_2, ..., d_p)$ with $P - 1 \ge d_1 \ge d_2 \ge ...... \ge d_p$ is graphical if and only if the following procedure results in a partition with every summand zero.

(i) Determine the modified partition π' as in the statement of the theorem.

(ii) Reorder the terms of π' so that they are non increasing and call the resulting partition π_1.

(iii) Determine the modified partition π'' of π_1 as in step (i) and the reordered partition π_2.

(iv) Continue the process as long as non negative summands can be obtained.

Theorem 1.13. *The complete graph K_{2n} is 1-factorable.*

Proof. We need only display a partition of the set X of lines of K_{2n} into $(2n - 1)$ 1-factors.

For this purpose we denote the points of G by

$v_1, v_2, ..., v_{2n}$ and define for $i = 1, 2, ..., 2n - 1$, the sets of lines

$$X_i = \{v_iv_{2n}\} \cup \{v_{i-j}\, v_{i+j}\, ;\, j = 1, 2, ..., n - 1\}$$

Where each of the subscripts $i - j$ and $i + j$ is expressed as one of the numbers 1, 2, ..., $2n - 1$ modulo $(2n - 1)$.

The collection $\{X_i\}$ is easily seen to give an appropriate partition of X, and the sum of the subgraphs G_i induced by X_i is a 1-factorization of K_{2n}.

Theorem 7.14. *The complete graph K_{2n} is the sum of a 1-factor and $n - 1$ spanning cycles.*

Theorem 7.15. *Every bridgeless cubic graph is the sum of a 1-factor and a 2-factor.*

Theorem 7.16. *A graph is 2-factorabe if and only if it is regular of even degree.*

Theorem 7.17. *If a 2-connected graph has a 1-factor then it has at least two different 1-factors.*

Theorem 7.18. *Every regular bigraph is 1-factorable.*

Theorem 7.19. *The graph K_{2n+1} is the sum of n spanning cycles.*

Proof. In order to construct n line-disjoint spanning cycles in K_{2n+1}, first label its points $v_1, v_2, \ldots\ldots, v_{2n+1}$. Then construct n paths P_i on the points $v_1, v_2, \ldots\ldots, v_{2n}$ as follows : $P_i = v_i\, v_{i-1}\, v_{i+1}\, v_{i-2} \ldots\ldots v_{i+n-1}\, v_{i-n}$.

Thus the j_{th} point of P_i is v_k, where $k = i + (-1)^{j+1}\,[j/2]$

and all subscripts are taken as the integers 1, 2, ..., $2n \pmod{2n}$

The spanning cycle z_i is then constructed by joining v_{2n+1} to the endpoints of P_i.

Theorem 7.20. *Let G be a given graph and let f be a function from V into the non negative integers. Then G has no spanning subgraph whose degree sequence is prescribed by f if and only if there exist disjoint sets S and T of points such that* $\sum_{u \in S} f(u) < k_0(S, T) + \sum_{V \in T}\left[f(v) - d_{G-S}(v)\right]$.

Theorem 7.21. *A graph G has a 1-factor if and only if P is even odd there is no set S of points such that the number of odd components of G – S exceeds |S|.*

Proof. Necessity part

Let S be any set of points of G and let H be a component of G – S.

In any 1-factor of G, each point of H must be paired with either another point of H or a point of S.

But if H has no odd number of points, then at least one point of H is matched with a point of S.

Let k_0 be the number of odd components of G – S.

If G has a 1-factor then $|S| \geq k_0$, since in a 1-factor each point of S can be matched with at most one points of G – S and therefore can take care of at most one odd component.

Sufficiency part

Assume that G does not have a 1-factor, and let S be a maximum set of independent lines.

Let T denote the set of lines not in S, and let u_0 be a point incident only with lines in T.

A trail is called alternating if the lines alternately lie in S and T.

For each point $v \neq u_0$, call v a 0-point if there are no $u_0 - v$ alternating trails, if there is such a trail, call v an S-point if all these trails terminate in a line of S at v, a T-point if each terminates in a line of T at v, and an ST-point if some terminate in each type of line.

The following statements are immediate consequences :

Every point adjacent to u_0 is a T-or an ST-point.

No S-or 0-point is adjacent to any S or ST-point.

No T-point is joined by a line of S to any T-or O-point

Therefore, each S-point is joined by a line of S to a T-point.

Furthermore, each T-point vx is incident with a line of S since otherwise the lines in an alternating $u_0 - v$ trail could be switched between S and T to obtain a larger independent set.

Let H be the graph obtained by deleting the T-points. One component of H contains u_0, and any other points in it are ST- points.

The other components wither consist of an isolated S-point, only ST-points, or only 0-points.

We now show that any component H_1 of H containing ST- points has an odd number of them.

Obviously H_1 either contains u_0 or has a point u_1 joined in G to a T-point by a line of S such that some alternating $u_0 - u_1$ trail contains this line and no other points of H_1.

If H_1 contains u_0, we take $u_1 = u_0$.

The following argument will be used to show that within H_1 every point v other than u_1 is incident with some line of S.

This is accomplished by showing that there is an alternating $u_1 - v$ trail in H_1 which terminates in a line of S.

The first step in doing this is showing that if there is an alternating $u_1 - v$ trail P_1, then there is one which terminates in a line of S. Let P_2 be an alternating $u_0 - v$ trail ending in a line of T, and let $u'v'$ be the last line of P_2, if any, which does not lie in H_1.

Then u' must be a T-point and $u'v'$ a line in S. Now go along P_1 from u_1 until a point w' of P_2 is reached.

Continuing along P_2 in one of the two directions must give an alternating trail.

If going to v' results in an alternating path, then the original $u_0 - u_1$ trail P_0 followed by this new path and the line $v'u'$ would be a $u_0 - u_1$ trial terminating in a line of S and u' could not be a T-point.

Hence there must be a $u_1 - v$ trail terminating in a line of S.

Now we show show that there is necessarily a $u_1 - v$ alternating trail by assuming there is not.

Then threre is a point w adjacent to v for which there is a $u_1 - w$ alternating trail.

If line wv is in S then the $u_1 - w$ alternating trail terminates in a line of T, while if wv is in T, the preceding argument shows there is a $u_1 - w$ trail terminating in a line of S. In either case, there is a $u_1 - v$ alternating trail.

This shows that the component H_1 has an odd number of points, and theat if H_1 does not contain u_0, exactly one of its points is joined to a T-point by a line of S.

Hence, with the exception of the component of H containing u_0 and those consisting entirely of 0-points, each is paired with exactly one T-point by a line in S. Since each of these and the component containing u_0 is odd, the theorem is proved.

Theorem 7.22. *Let G be a nontrivial (p, q) graph and let q_n be the maximum number of lines in any subgraph of G having n points. Then*

$$r(G) = max_n \left\{ \frac{q_n}{n-1} \right\}.$$

Corollary. The arboricities of the complete graphs and bigraphs are $r(k_P) = \left\{ \frac{P}{2} \right\}$ and

$$r(k_{r,\, s}) = \left\{ \frac{rs}{r+s-1} \right\}.$$

Problem Set 7.1

1. Prove that a covering g of a graph is minimal if and only if g contains no paths of length three or more.

2. Explore how the covering number of a graph G with n vertices is related to the diameter of G.

3. Sketch a graph with an even number of vertices that has no dimer covering.

4. Prove that any nontrivial connected graph G.

$$\alpha_0 + \beta_0 = P = \alpha_1 + \beta_1.$$

5. If G is bipartite then show that the number of lines in a maximum matching equals the point covering number, that is $\beta_1 = \alpha_0$.

6. If Y is a line cover of G such that there is no Y-reducing walk then show that Y is a minmum line cover.

7. Prove that a point v is critical in a graph G if and only if some minimum point cover contains v.

8. Prove that, the following are equivalent for any graph G

(*i*) G has a line-core

(*ii*) G has an external minimum point cover

(*iii*) Every minimum point cover for G is external.

9. Prove or disprove : A line x is critical in a graph G if and only if there is a minimum line cover containing x.

10. Prove or disprove : Every point cover of a graph G contains a minimum point cover.

11. If G has a closed trail containing a point cover then show that L(G) is hamiltonian.

12. Calculate

(*i*) α_{11} (K_P) (*ii*) α_{00} ($K_{m,\,n}$) (*iii*) α_{11} ($K_{m,n}$)

13. If G is regular of degree n then show that there is a partition of V into at most 1 + [n/2] subsets such that each point is adjacent to at most one other point in the same subset.

14. Prove or disprove : Every 2-connected line-critical graph is hamiltonian.

15. If G is a connected graph having a line-core c_1 (G) then show that

(*i*) C_1 (G) is a spanning subgraph of G

(*ii*) C_1 (C_1(G)) = C_1 (G)

(*iii*) The components of C_1 (G) are bipartite subgraphs of G which are not reducible.

16. Prove that the complete graph K_{2n} is 1-factorable.

17. Let G be a given graph and let f be a function from V into the nonnegative integers, then show that G has no spanning subgraph whose degree sequence is prescribed by f if and obly if there exist disjoint sets S and T of points such that $\sum_{u \in S} f(u) < k_0(S, T) + \sum_{v \in T} \left[f(v) - d_{G-S}(v) \right]$.

18. Prove that every bridgeless cubic graph is the sum of a 1-factor and a 2-factor.

19. Prove that a graph is 2-factorable if and only if it is regular of even degree.

20. Prove or disprove : Let G be a graph with a 1-factor F. A line of G is in more than one 1-factor if and only if it lies on a cycle whose lines are alternately in F.

21. The graph K_4 has a unique 1-factorization. Find the number of 1-factorizations of $K_{3,\,3}$ and of K_6.

22. Find the smallest connected (p, q) graph G such that

$$\max_r \{q_r/(r-1)\} > \{q/(p-1)\}$$

where q_r is the maximum number of lines in any induced subgraph of G with r points.

23. If an n-connected graph G with P even is regular of degree n then show that G has a 1-factor.

24. Display a minimal decomposition of $K_{4,4}$ into spanning forests.

25. If a 2-connected graph has a 1-factor then show that it has at least two different 1-factor.

26. Prove that every regular bigraph is 1-factorable.

27. Prove that the complete graph K_{2n} is the sum of a 1-factor and $n - 1$ spanning cycles.

28. Prove that the graph K_{2n+1} is the sum of n spanning cycles.

29. Verify that $\{v_1, v_2, v_6, v_8\}$ is a minimal vertex covering in the following graph.

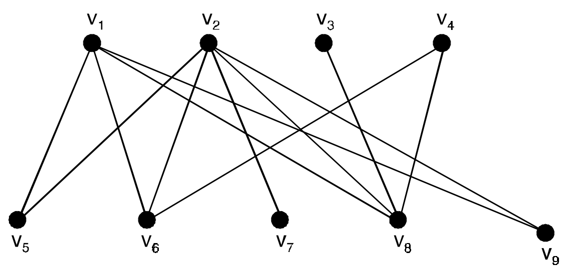


Fig. 7.7.

30. Prove that every pendant edge of a graph must belong to every edge covering of the graph.

31. Prove that the number of veritces in a vertex covering of a graph is greater than or equal to the number of edges in every matching of the graph.

32. Disprove that a pendant vertex of a graph must belong to a vertex covering of the graph.

33. For the graph shown below, verify that $\{v_1, v_3, v_4, v_5\}$ and $\{v_2, v_4, v_5, v_6\}$ are minimal vertex coverings and $\{e_1, e_4, e_8\}$ and $\{e_2, e_4, e_6\}$ are minimal edge coverings.

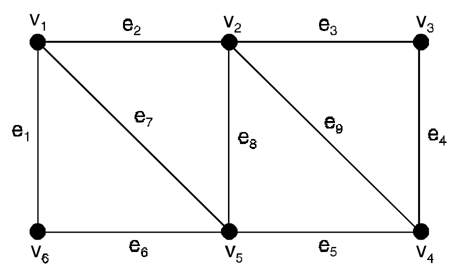


Fig. 7.8.

34. Prove that a subset W of the vertex set V of a graph G is a vertex covering of G if and only if no two veritces in the subgraph V – W are adjacent.

35. Prove that an edge covering S of a graph G is minimal if and only if S contains no pahs of length three or more.

CHAPTER

8 Digraphs

8.1 DIGRAPH DEFINITION

A digraph D consists of a finite set V of points and a collection of ordered pairs of distinct points. Any such pair (u, v) is called an arc or directed line and will usually be denoted uv. The arc uv goes from u to v and is incident with u and v. We say that u is adjacent to v and v is adjacent from u.

In otherwords, A directed graph or a digraph G consists of a set of vertices $V = \{v_1, v_2, ...\}$, a set of edge $E = \{e_1, e_2, ...\}$ and a mapping ψ that maps every edge onto some ordered pair of vertices (v_i, v_j).

For example, Fig. 8.1(a) below shows a digraph with five vertices and ten edges.

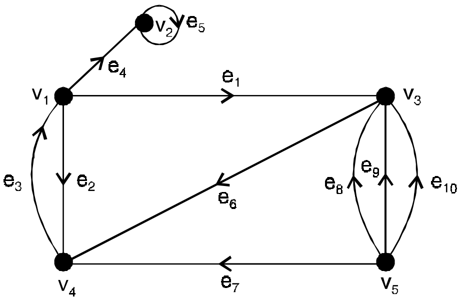
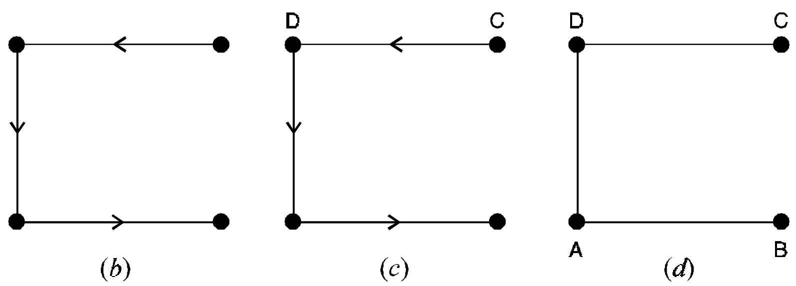


Fig. 8.1.(*a*) Directed graph with 5 vertices and 10 edges.

8.2 ORIENTATION OF A GRAPH

Given a graph G, if there is a digraph D such that G is the underlying graph of D then D is called an orientation of G.

The digraphs in Fig. 8.1(b) and Fig. 8.1(c) are two different orientations of the graph in Fig. 8.1(d).



Fog. 8.1(*b*), (*c*), (*d*).

8.3 UNDERLYING GRAPH

If D is a digraph, the graph obtained from D by 'removing the arrows' from the directed edges is called the underlying graph of D. This graph is also called the undirected graph corresponding to D.

The underlying graph of the digraph in Fig. 8.1(*b*) is shown in Fig. 8.1(*d*).

The graph in Fig. 8.1(*d*) is the underlying graph of the digraph shown in Fig. 8.1(*c*).

Note : Every digraph has a unique underlying graph.

8.4 PARALLEL EDGES

Two (directed) edges e and e' of a digraph D are said to be parallel if e and e' have the same initial vertex and the same terminal vertex.

In the digraph in Fig. 8.2 the edges e_6 and e_7 are parallel edges whereas the edges e_1 and e_9 are not parallel. e_1 and e_6 are parallel edges in the underlying graph.

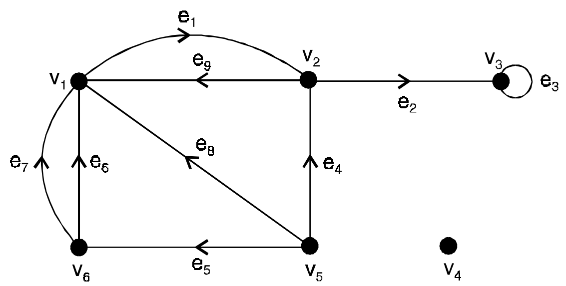


Fig. 8.2.

8.5 INCIDENCE

In a digraph every edge has two end vertices, one vertex from which it begins and the other vertex at which it terminates. If an edge e begins at a vertex u and terminates at a vertex v, we say that e is incident out of u and incident into v. Here, u is called the initial vertex and v is called the terminal vertex of e.

For example, in the digraph in Fig. 8.2, the edge e_1 is incident out of the vertex v_1 and incident into the vertex v_2, v_1 is the initial vertex and v_2 is the terminal vertex of the edge e_1. For a self-loop in a digraph, the initial and terminal vertices are one and the same. In Fig. 8.2 the edge e_3 is a self-loop with v_3 as the initial and terminal vertex.

8.6 IN-DEGREE AND OUT-DEGREE

If v is a vertex of a digraph D, the number of edges incident out of v is called the out-degree of v and the number of edges incident into v is called the in-degree of v. The out-degree of v is denoted by $d^+(v)$ and the in-degree of v is denoted by $d^-(v)$.

For example, the out-degrees and in-degrees of the six vertices of the digraph shown in Fig. 8.2. are as given below :

$d^+(v_1) = 1, \quad d^-(v_1) = 4$
$d^+(v_2) = 2, \quad d^-(v_2) = 2$
$d^+(v_3) = 1, \quad d^-(v_3) = 2$
$d^+(v_4) = 0, \quad d^-(v_4) = 0$
$d^+(v_5) = 3, \quad d^-(v_5) = 0$
$d^+(v_6) = 2, \quad d^-(v_6) = 1$

For example, in Fig. 8.1(*a*)

$d^+(v_1) = 3, \quad d^-(v_1) = 1$
$d^+(v_2) = 1, \quad d^-(v_2) = 2$
$d^+(v_5) = 4, \quad d^-(v_5) = 0.$

8.7 ISOLATED VERTEX

If v is a vertex of a digraph D then v is called an isolated vertex of D if $d^+(v) = d^-(v) = 0$.

8.8 PENDANT VERTEX

If v is a vertex of a digraph D then v is called a pendant vertex of D if $d^+(v) + d^-(v) = 1$.

8.9 SOURCE

If v is a vertex of a digraph D then v is called a source of D if $d^-(v) = 0$.

8.10 SINK

If v is a vertex of a digraph D then v is called a sink of D if $d^+(v) = 0$.

The digraph is Fig. 8.1(*a*) has v_4 as an isolated vertex and v_5 as a source.

In the digraph in Fig. 8.1(*a*), the vertices B and C are pendant vertices, C is a source and B is a sink.

8.11 TYPES OF DIGRAPHS

8.11.1. Simple Digraphs

A digraphs that has no self-loop or parallel edges is called a simple digraph.

The digraph shown in Fig. 8.3(*a*) is simple, but its underlying graph shown in Fig. 8.3(*b*) is not simple.

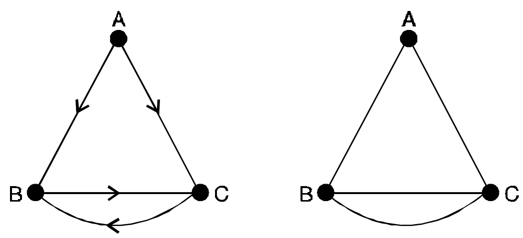


Fig. 8.3.(*a*), (*b*).

8.11.2. A Symmetric Digraphs

Digraphs that have atmost one directed edge between a pair of vertices, but are allowed to have self-loops, are called asymmetric or antisymmetric digraph.

For example, the digraph in Fig. 8.4(*a*), is asymmetric.

The digraph in Fig. 8.4(*b*) is neither symmetric nor asymmetric.

The digraph in Fig. 8.4(*b*) is simple and asymmetric.

The digraph in Fig. 8.4(*b*) is simple but not asymmetric.

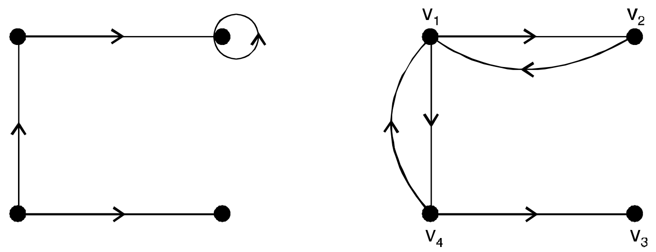


Fig. 8.4.(*a*), (*b*).

8.11.3. Symmetric Digraph

Digraphs in which for every edge (*a*, *b*) (*i.e.*, from vertex *a* to *b*) there is also an edge (*b*, *a*).

For example, the digraph in Fig. 8.5 is a symmetric digraph. The digraph in Fig. 8.4(*b*) is not symmetric.

This digraph has (v_4, v_3) as an edge but does not have (v_3, v_4) as an edge.

The digraph in Fig. 8.5 is simple also. Such a digraph is called a symmetric simple digraph. The digraph in Fig. 8.4(*b*) is simple and non-symmetric.

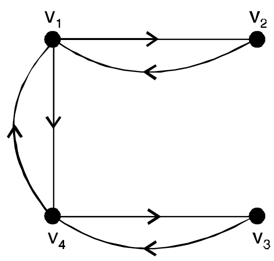


Fig. 8.5.

8.11.4. Isomorphic Digraphs

Isomorphic graphs were defined such that they have identical behaviour in terms of graph properties.

In otherwords, if their labels are removed, two isomorphic graphs are indistinguishable. For two digraphs to be isomorphic not only must their corresponding undirected graphs be isomorphic, but the directions of the corresponding edges must also agree.

For example, Fig. 8.6, shows two digraphs that are not isomorphic, although they are orientations of the same undirected graph.

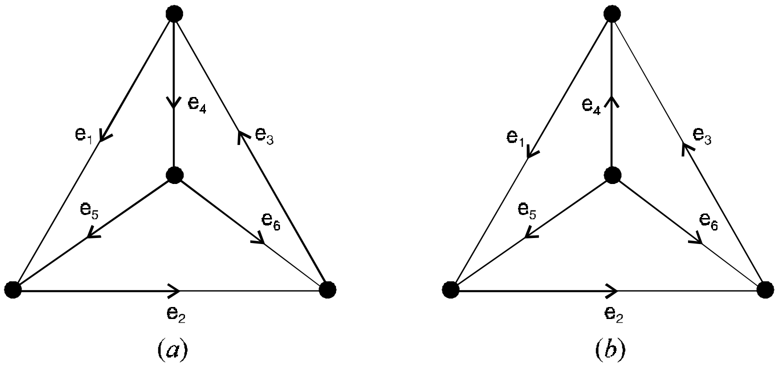


Fig. 8.6. Two nonisomorphic digraphs.

In otherwords, two digraphs D_1 and D_2 are said to be isomorphic if both of the following conditions hold :

(*i*) The underlying graphs of D_1 and D_2 are either identical or isomorphic.

(*ii*) Under the one-to-one correspondence between the edges of D_1 and D_2 the directions of the corresponding edges are preserved.

The two digraphs in Fig. 8.7(*a*) and 8.7(*b*) are isomorphic, whereas the two digraphs in Fig.8.8(*a*) and 8.8(*b*) are not isomorphic.

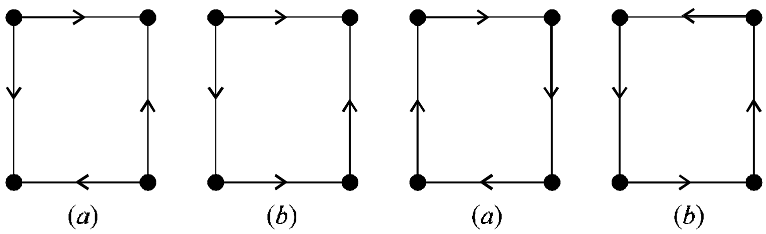


Fig. 8.8. Two isomorphic digraphs. **Fig. 8.8. Two non-isomorphic digraphs.**

8.11.5. Balanced Digraphs

A digraph D is said to be a balanced digraph or an isograph if $d^+(v) = d^-(v)$ for every vertex v of D.

8.11.6. Regular Digraph

A balanced digraph is said to be regular if every vertex has the same in-degree and out-degree as every other vertex.

8.11.7. Complete Digraphs

A complete undirected graph was defined as a simple graph in which every vertex is joined to every other vertex exactly by one edge.

8.11.8. Complete Symmetric Digraph

A complete symmetric digraph is a simple digraph in which there is exactly one edge directed from every vertex to every other vertex (see Fig. 8.9).

8.11.9. Complete Asymmetric Digraph

A complete asymmetric digraph is an asymmetric digraph in which there is exactly one edge between every pair of vertices (see Fig. 8.6).

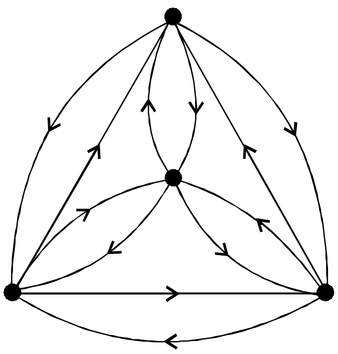


Fig. 8.9. Complete Symmetric Digraph of Four Vertices.

8.12 CONNECTED DIGRAPHS

8.12.1. Strongly Connected

A digraph G is said to be strongly connected if there is atleast one directed path from every vertex to every other vertex.

8.12.2. Weakly Connected

A digraph G is said to be weakly connected if its corresponding undirected graph is connected but G is not strongly connected.

Fig. 8.6, one of the digraphs is strongly connected, and the other one is weakly connected.

8.12.3. Component and Fragments

Each maximal connected (weakly or strongly) subgraph of a digraph G is called a component of G. But within each component of G the maximal strongly connected subgraphs are called the fragments (or strongly connected fragments) of G.

For example, the digraph in Fig. 10, consists of two components. The component g_1 contains three fragments $\{e_1, e_2\}$, $\{e_5, e_6, e_7, e_8\}$ and $\{e_{10}\}$.

We observe that e_3, e_4 and e_9 do not appear in any fragment of g_1.

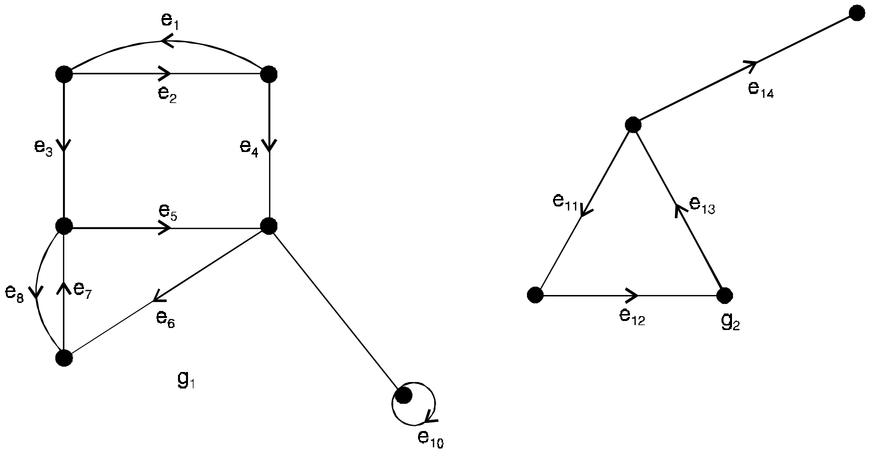


Fig. 8.10. Disconnected digraph with two components.

8.13 CONDENSATION

The condensation G_c of a digraph G is a digraph in which each strongly connected fragment is replaced by a vertex and all directed edges from one strongly connected component to another are replaced by a single directed edge.

The condensation of the digraph G in Fig. 8.10 is shown in Fig. 8.11.

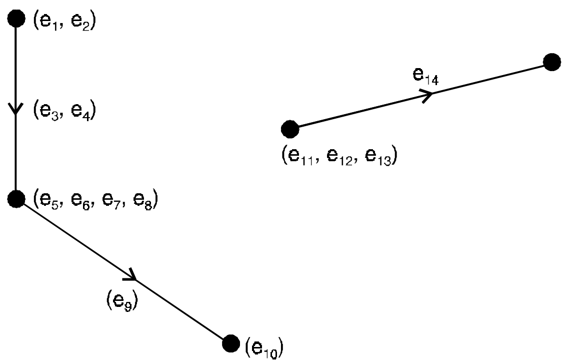


Fig. 8.11. Condensation of Fig. 8.10.

Observations :

(*i*) The condensation of a strongly connected digraph is simply a vertex.

(*ii*) The condensation of a digraph has no directed circuit.

8.14 REACHABILITY

Given two vertices *u* and *v* of a digraph D, we say that *v* is reachable (or accessible) from *u* if there exists atleast one directed path in D from *u* to *v*.

For example, in the digraph shown in Fig. 8.1, the vertex v_3 is reachable from the vertex v_5, but v_5 is not reachable from v_3.

8.15 ORIENTABLE GRAPH

A graph G is said to be orientable if there exists a strongly connected digraph D for which G is the underlying graph.

For example, the graph is Fig. 8.12(*a*) is orientable, a strongly directed digraph for which this graph is the underlying graph is shown in Fig. 8.12(*b*).

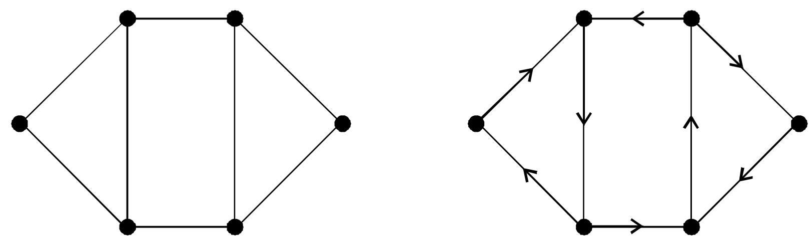


Fig. 8.12.(*a*) (*b*)

8.16 ACCESSIBILITY

In a digraph a vertex *b* is said to be accessible (or reachable) from vertex *a* if there is a directed path from *a* to *b*. Clearly, a digraph G is strongly connected if and only if every vertex in G is accessible from every other vertex.

8.17 ARBORESCENCE

A digraph G is said to be an arborescence if

(*i*) G contains no circuit, neither directed nor semi circuit.

(*ii*) In G there is precisely one vertex *v* of zero in-degree.

This vertex *v* is called the root of the arborescence.

An arborescence is shown in Fig. 8.13 below.

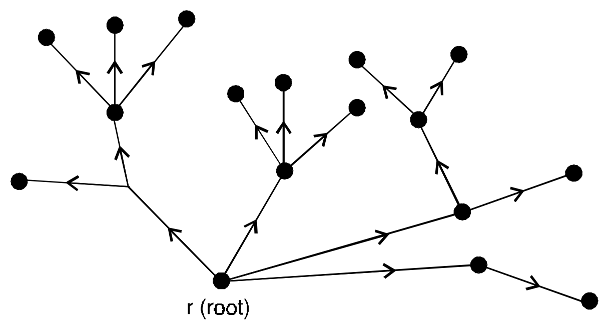


Fig. 8.13. Arborescence.

8.17.1. Spanning arborescence

A spanning tree in an *n*-vertex connected digraph, analogous to a spanning tree in an undirected graph, consists of *n* – 1 directed edges.

A spanning arborescence in a connected digraph is a spanning tree that is an arborescence.

For example, a spanning arborescence in Fig. 8.14, is {*f*, *b*, *d*}. There is a striking relationship between a spanning arborescence and an Euler line.

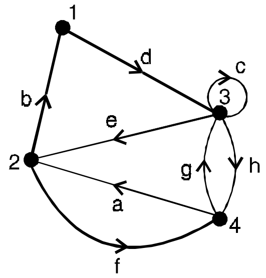


Fig. 8.14. Euler Digraph.

8.18 EULER DIGRAPHS

In a digraph G a closed directed walk (*i.e.*, a directed walk that starts and ends at the same vertex) which transverses every edge of G exactly once is called a directed Euler line.

A graph containing a directed Euler line is called an Euler digraph.

For example, the graph in Fig. 8.15, is an Euler digraph, in which the walk *a b c d e f* is an Euler line.

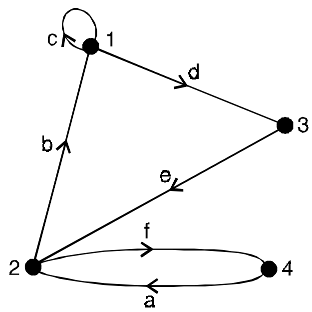


Fig. 8.15. Euler Digraph.

8.19 HAND SHAKING DILEMMA

In a digraph D, the sum of the out-degree of all vertices is equal to the sum of the in-degrees of all vertices, each sum being equal to the numbe of edges in D.

For example, the digraph in Fig. 8.1, we note that the digraphs has 6 vertices and 9 edges and that the sums of the out-degrees and in-degrees of its vertices are

$$\sum_{i=1}^{6} d^{+}(v_i) = 9 \; ; \; \sum_{i=1}^{6} d^{-}(v_i) = 9$$

8.20 DIRECTED WALK, DIRECTED PATH, DIRECTED CIRCUIT

8.20.1. Directed walk

A directed walk or a directed trail in D is a finite sequence whose terms are alternately vertices and edges in D such that each edge is incident out of the vertex preceeding it in the sequence and incident into the vertex following it.

A directed walk or a directed trail in D is a sequence of the form $v_0\ e_1\ v_1\ e_2\ .....\ e_k\ v_k$ where $v_0, v_1,$ ...... v_k are vertices of D in some order and $e_1, e_2,\ .....\ e_k$ are edges of D such that the edge e_i has v_{i-1} as the initial vertex and v_i as the terminal vertex, $i = 1, 2,\ .....\ k$.

A vertex can appear more than once in a directed walk but not an edge.

The vertex with which a directed walk begins is called its initial vertex and the vertex with which its ends is called its final or terminal vertex.

8.20.2. Directed path

An open directed walk in which no vertex is repeated is called a directed path.

8.20.3. Directed circuit

A closed directed walk in which no vertices, except the initial and final vertices are repeated is called a directed circuit or a directed cycle.

8.20.4. Length

The number of edges present in a directed walk, directed path, directed circuit is called its length.

For example, in the digraph shown in Fig. (8.1)

(*i*) $v_1e_1v_2e_2v_3e_3v_3$ is an open directed walk which is not a directed path, its length is 3.

(*ii*) $v_6e_6v_1e_1v_2e_2v_3$ is an open directed walk which is a directed path, its length is 3.

(*iii*) $v_1e_1v_2e_9v_1$ or $v_1v_2v_1$ is a closed directed walk which is a directed circuit, its length is 2.

8.21 SEMI-WALK, SEMI-PATH, SEMI-CIRCUIT

8.21.1. Semi-Walk

A semi-walk in a digraph D is a walk in the underlying graph of D, but is not a directed walk in D. A walk in D can mean either a directed walk or a semi-walk in D.

8.21.2. Semi-path

A semi-path in a digraph D is a path in the underlying graph of D, but is not a directed path in D. A path in D can mean either a directed path or a semi-path in D.

8.21.3. Semi-circuit

A semi-circuit in a digraph D is a circuit in the underlying graph of D, but is not a directed circuit in D. A circuit in D can mean either a directed circuit or a semi-circuit in D.

For example, in the digraph in Fig. (8.1), the sequence $v_6e_6v_1e_9v_2e_4v_5$ is a semi-path and the sequence $v_5e_5v_2e_1v_1e_8v_5$ is a semi-circuit.

8.22 TOURNAMENTS

A tournament is an oriented complete graph. All tournaments with two, three and four points are shown in Fig. 8.16.

The first with three points is called a transitive triple, the second a cycle triple.

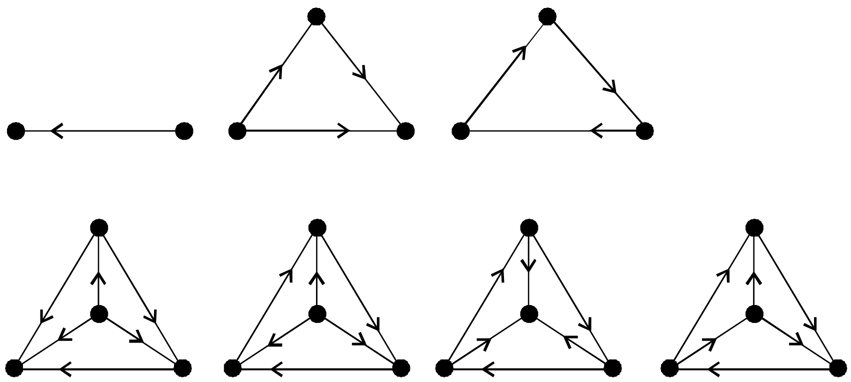


Fig. 8.16. Small tournament.

8.23 INCIDENCE MATRIX OF A DIGRAPH

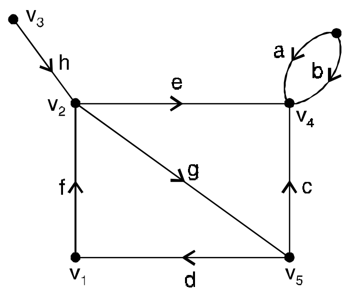
The incidence matrix of a digraph with n vertices, e edges and no self-loops in an n by n matrix A = $[a_{ij}]$ whose rows correspond to vertices and columns correspond to edges such that

a_{ij} = 1, if j^{th} edge is incident out of i^{th} vertex

= – 1, if j^{th} edge is incident into i^{th} vertex

= 0, if j^{th} edge is not incident on i^{th} vertex.

For example, A digraph and its incidence matrix are shown in Fig. 8.17.



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array}
\begin{array}{c}
\begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
-1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & -1 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}$$

Fig. 8.17. Digraph and its incidence matrix.

8.24 CIRCUIT MATRIX OF A DIGRAPH

Let G be a digraph with e edges and q circuits. An arbitrary orientation is assigned to each of the q circuits. Then a circuit matrix B = $[b_{ij}]$ of the digraph G is a q by e matrix defined as

b_{ij} = 1, if i^{th} circuit includes j^{th} edge, and the orientations of the edge and circuit coincide
= – 1, if i^{th} circuit includes j^{th} edge, but the orientations of the two are opposite
= 0, if i^{th} circuit does not include the j^{th} edge.

For example, a circuit matrix of the digraph in Fig. 8.17 is

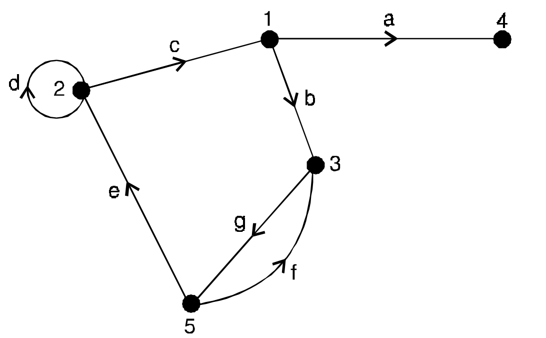
$$\begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 & -1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

8.25 ADJACENCY MATRIX OF A DIGRAPH

Let G be a digraph with n vertices, containing no parallel edges. Then the adjacency matrix X = [x_{ij}] of the digraph G is an n by n (0, 1) matrix whose element.

x_{ij} = 1, if there is an edge directed from i^{th} vertex to j^{th} vertex
= 0, otherwise

For example, a digraph and its adjacency matrix are shown in Fig. 8.18.



$$X = \begin{array}{cc} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

Fig. 8.18. Digraph and its Adjacency Matrix.

Observations :

(*i*) X is a symmetric matrix if and only if G is a symmetric digraph.

(*ii*) Every non-zero element on the main diagonal represents a self-loop at the corresponding vertex.

(*iii*) There is no way of showing parallel edges in X. This is why the adjacency matrix is defined only for a digraph without parallel edges.

(*iv*) The sum of each row equals the out-degree of the corresponding vertex and the sum of each column equals the in-degree of the corresponding vertex. The number of non-zero entries in X equals the number of edges in G.

(v) If X is the adjacency matrix of a digraph G, then the transposed matrix X^T is the adjacency matrix of a digraph G^R obtained by reversing the direction of every edge in G.

(vi) For any square (0, 1)-matrix Q of order n, there exists a unique digraph G of n vertices such that Q is the adjacency matrix of G.

Theorem 8.1. *Let G be a connected graph. Then G is orientable if and only if each edge of G is contained in at least one cycle.*

Proof. The necessity of the condition is clear. To prove the sufficiency.

We choose any cycle C and direct its edges cyclically.

If each edge of G is contained in C, then the proof is complete. If not, we choose any edge e that is not in C but which is adjacent to an edge of C.

By hypothesis, e is contained in some cycle C′ whose edges we may direct cyclically, except for those edges that have already been directed, that is, those edges of C′ that also lie in C.

It is not difficult to see that the resulting digraph is strongly connected, the situation is illustrated in Fig. 8.19 below, with dashed lines denoting edges of C′.

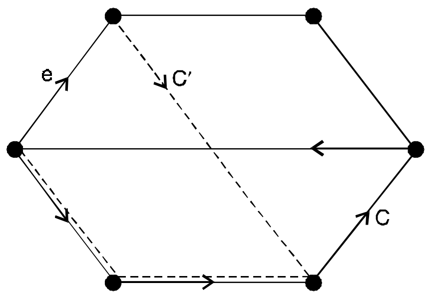


Fig. 8.19.

We proceed in this way, at each stage directing at least one new edge, until all edges are directed. Since the digraph remains strongly connected at each stage, the result follows.

Theorem 8.2. *A connected digraph is Eulerian if and only if for each vertex of D out deg(V) = in deg(v).*

Theorem 8.3. *Let D be a strongly connected digraph with n vertices. If out deg (v)* $\geq \frac{n}{2}$ *and in deg (v)* $\geq \frac{n}{2}$ *for each vertex v, then D is Hamiltonian.*

Theorem 8.4. (i) *Every non-Hamiltonian tournament is semi-Hamiltonian,*

(ii) *every strongly connected tournament is Hamiltonian.*

Proof. (i) The statement is clearly true if the tournament has fewer than four vertices.

We prove the result by induction on the number of vertices.

Assume that every non-Hamiltonian tournament on n vertices is semi-Hamiltonian.

Let T be a non-Hamiltonian tournament on $n + 1$ vertices, and let T′ be the tournament on n vertices obtained by removing from T a vertex v and its incident arcs.

By the induction hypothesis, T′ has a semi-Hamiltonian path $v_1 \to v_2 \to \ldots\ldots \to v_n$.

There are now three cases to consider

(1) if vv_1 is an arc in T, then the required path is $v \to v_1 \to v_2 \ \ldots\ldots\ldots \to v_n$.

(2) if vv_1 is not an arc in T, which means that v_1v is and if there exists an i such that vv_i is an arc in T, then choosing i to be the first such, the required path is (see Fig. 8.20(a) below)

$$v_1 \to v_2 \to \ldots\ldots \to v_{i-1} \to v \to v_i \to \ldots\ldots \to v_n.$$

(3) if there is no arc in T of the form vv_i, then the required path is $v_1 \to v_2 \to \ldots\ldots \to v_n \to v$.

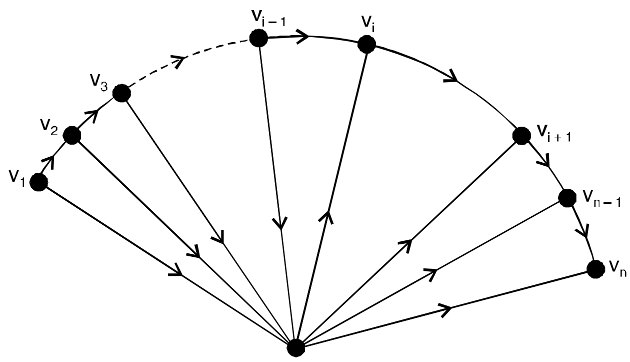


Fig. 8.20(*a*).

(ii) We prove the stronger result that a strongly connected tournament T on n vertices contains cycles of length 3, 4, ... n.

To show that T contains a cycle of length 3.

Let v be any vertex of T and let W be the set of all vertices W such that vW is an arc in T, and z be the set of all vertices z such that zv is an arc.

Since T is strongly connected W and Z must both be non-empty, and there must be an arc in T of the form $w'z'$, where w' is an W and z' is in Z (see Fig. 8.20(b) below). The required cycle of length 3 is then $v \to \omega' \to z' \to v$.

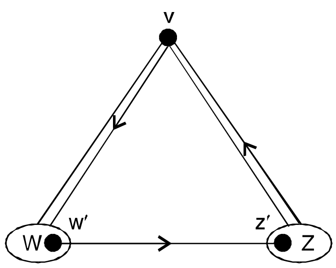


Fig. 8.20(*b*).

It remains only to show that, if there is a cycle of length k, where $k \le n$, then there is one of length $k + 1$.

Let $v_1 \to ...... v_k \to v_1$ be such that a cycle.

Suppose first that there exists a vertex v not contains in this cycle, such that there exist arcs in T of the form vv_i and of the form $v_j v$.

Then there must be a vertex v_i such that both $v_{i-1}v$ and vv_i are arcs in T. The required cycle is then $v_1 \to v_2 \to .... \to v_{i-1} \to v \to v_i \to ...... \to v_k \to v_1$ (see Fig. 8.20(c)).

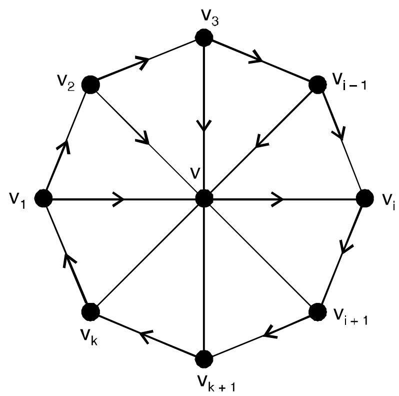


Fig. 8.20.(c)

If no vertex exists with the above-mentioned property, then set of vertices not contained in the cycle may be divided into two disjoint sets W and Z, where W is the set of vertices w such that vw_i is an arc for each i, and Z is the set of vertices z such taht zv_i is an arc for each i.

Since T is strongly connected, W and Z must both be non-empty, and there must be an arc in T of the form $w'z'$, where w' is in W and z' is in Z.

The required cycle is then $v_1 \to w' \to z' \to v_3 \to ...... \to v_k \to v_1$. (See Fig. 8.20($d$) below).

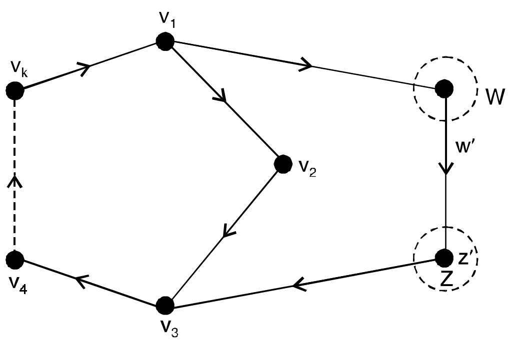


Fig. 8.20(d).

Theorem 8.5. *A digraph is strong if and only if it has a spanning closed walk, it is unilateral if and only if it has a spanning walk, and it is weak if and only if it has a spanning semi-walk.*

Theorem 8.6. *A weak digraph is an in-tree if and only if exactly one point has out degree 0 and all others have out degree 1.*

Theorem 8.7. *A weak digraph is an out-tree if and only if exactly one point has indegree 0 and all others have indegree 1.*

Theorem 8.8. *Every digraph with no odd cycles has a 1-basis.*

Corollary. Every acylic digraph has a 1-basis.

Theorem 8.9. *Every acyclic digraph has a unique point basis consisting of all points of indegree 0.*

Corollary. Every point basis of a digraph D consists of exactly one point from each of those strong components in D which form the point basis of D*.

Theorem 8.10. *An cyclic digraph D has at least one point of indegree zero.*

Theorem 8.11. *An acyclic digraph has at least one point of out degree zero.*

Proof. Consider the last point of any maximal path in the digraph. This point can have no points adjacent from it since otherwise there would be a cycle or the path would not be maximal.

The dual theorem follows immediately by applying the principle of Directional Duality. In keeping with the use of D′ to denote the converse of digraph D.

Theorem 8.12. *The following properties of a digraph D are equivalent.*

(*i*) D is a acyclic.

(*ii*) D* is isomorphic to D.

(*iii*) Every walk of D is a path.

(*iv*) It is possible to order the points of D so that the adjacency matrix A(D) is upper triangular.

Theorem 8.13. *The following are equivalent for a weak digraph D.*

(*i*) *D is functional.*

(*ii*) *D has exactly one cycle, the removal of whose arcs results in a digraph in which each weak component is an in-tree with its sink in the cycle.*

(*iii*) *D has exactly one cycle z, and the removal of any arc of Z results in an in-tree.*

Problem 8.1. *Teleprinter's Problem*

How long is a longest circular (or cycle) sequence of 1's and 0's such that no subsequence of r bits appears more than once in the sequence ? Construct one such longest sequence.

Solution. Since there are 2^r distinct r-tuples formed from 0 and 1, the sequence can be no longer than 2^r bits long. We shall construct a circular sequence 2^r bits long with the required property that no subsequence of r bits be repeated.

Construct a digraph G whose vertices are all $(r - 1)$ tuples of 0's and 1's.

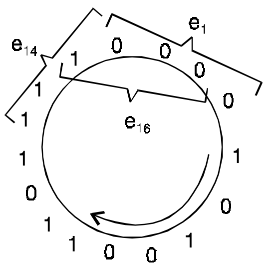
Clearly, there are 2^{r-1} vertices in G.

Let a typical vertex be $\alpha_1\alpha_2 \ldots\ldots \alpha_{r-1}$, where $\alpha_i = 0$ or 1.

Draw an edge directed from this vertex $(\alpha_1\alpha_2 \ldots\ldots \alpha_{r-1})$ to each of two vertices $(\alpha_2\alpha_3 \ldots\ldots \alpha_{r-1}0)$ and $(\alpha_2\alpha_3 \ldots\ldots \alpha_{r-1}1)$ label these directed edges $\alpha_1\alpha_2 \ldots\ldots \alpha_{r-1}0$ and $\alpha_1\alpha_2 \ldots\ldots \alpha_{r-1}1$ respectively.

Draw two such edges directed from each of the 2^{r-1} vertices. A self-loop will result in each of the two cases when $\alpha_1 = \alpha_2 = ... = \alpha_{r-1} = 0$ or 1.

The resulting digraph is an Euler digraph because for each vertex the in-degree equals the out-degree (each being equal to two). A directed Euler line in G consists of the 2^r edges, each with a distinct r-bit label. The labels of any two consecutive edges in the Euler line are of the form $\alpha_1\alpha_2 ...... \alpha_{r-1}\alpha_r$, $\alpha_2\alpha_3 ......\alpha_r\ \alpha_{r-1}$ that is ; the $r-1$ trailing bits of the first edge are identical to the $r-1$ leading bits of the second edge. Thus in the sequence of 2^r bits, made of the first bit of each of the edges in the Euler line, every possible subsequence of r bits occurs as the label of an edge, and since no two edges have the same label, no subsequence occurs more than once. The circular arrangement is achieved by joining the two ends of the sequence.



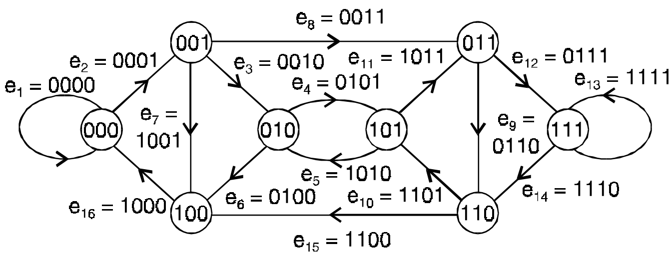


Fig. 8.21. Euler digraph for maximum-length sequence.

For $r = 4$, the graph in Fig. 8.21 above, illustrates the procedure of obtaining such a maximum length sequence one such sequence is 0000101001101111. Corresponding to the walk $e_1e_2e_3e_4e_5e_6e_7e_8e_9e_{10}\ e_{11}e_{12}e_{13}e_{14}e_{15}e_{16}$.

Problem 8.2. *Find the in-degrees and out-degrees of the vertices of the digraphs shown in Fig. 8.22 below. Also, verify the handshaking dilemma.*

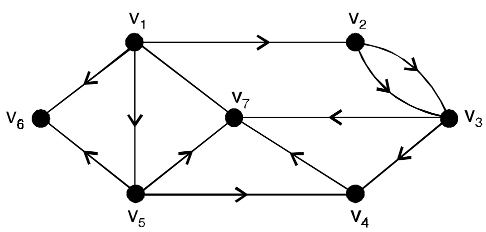


Fig. 8.22.

Solution. The given digraph has 7 vertices and 12 edges. The out-degree of a vertex is got by counting the number of edges that go out of the vertex and the in-degree of a vertex is got by counting the number of edges that end at the vertex. Thus, we obtain the following data.

Vertex	*Out-degree*	*In-degree*
v_1	4	0
v_2	2	1
v_3	2	2
v_4	1	2
v_5	3	1
v_6	0	2
v_7	0	4

This table gives the out-degrees and in-degrees of all vertices. We note that v_1 is a source and v_6 and v_7 are sinks.

Also, check that

sum of out-degrees = sum of in-degrees

= 12 = No. of edges.

Problem 8.3. *Let D be a digraph with an odd number of vertices prove that if each vertex of D has an odd out-degree then D has an odd number of vertices with odd in-degree.*

Solution. Let $v_1, v_2, ... v_n$ be the n vertices of D, where n is odd. Also let m be the number of edges in D.

They by handshaking dilemma

$$d^+(v_1) + d^+(v_2) + ... + d^+(v_n) = m \quad ...(1)$$

$$d^-(v_1) + d^-(v_2) + ... + d^-(v_n) = m \quad ...(2)$$

If each vertex v_i has odd out-degree, then the left hand side of (1) is a sum of n odd numbers. Since n is odd, this sum must also be odd. Thus m is odd.

Let k be the number of vertics with odd in-degree. Then $n - k$ number of vertices have even in-degree. Without loss of generality, let us take $v_1, v_2, ..., v_k$ to be the vertices with odd in-degree and $v_{k+1}, v_{k+2}, ... v_n$ to be the vertices with even in-degree.

Then, (2) may be rewritten as

$$\sum_{i=1}^{k} d^-(v_i) + \sum_{i=k+1}^{n} d^-(v_i) = m \quad ...(3)$$

Now the second sum on the left hand side of this expression is even. Also, m is odd. Therefore, the first sum must be odd. That is, $d^-(v_1) + d^-(v_2) + ... + d^-(v_k) = \text{odd}$ ...(4)

But, each of $d^-(v_1), d^-(v_2), ... d^-(v_k)$ is odd.

Therefore, the number of terms in the left hand side of (4) must be odd, that is ; k is odd.

Theorem 8.14. *A digraph G is an Eulerian digraph if and only if G is connected and is balanced that is ;* $d^-(v) = d^+(v)$ *for every vertex v in G.*

Theorem 8.15. *An arborescence is a tree in which every vertex other than the root has an in-degree of exactly one.*

Proof. An arborescence with n vertices can have at most $n - 1$ edges because of condition (1).

Therefore, the sum of in-degree of all vertices in G

$$d^-(v_1) + d^-(v_2) + ... + d^-(v_n) \le n - 1.$$

Of the n terms on the left-hand side of this equation, only one is zero because of condition (2), others must all be positive integers.

Therefore, they must all be 1's. Now since there are exactly $n - 1$ vertices of in-degree one and one vertex of in-degree zero, digraph G has exactly $n - 1$ edges. Since G is also circuitless, it must be connected, and hence a tree.

Example : In Fig. 8.23 below, W = (b d c e f g h a) is an Eulerian, starting and ending at vertex 2. The subdigraph {b, d, f} is a spanning arborescence rooted at vertex 2.

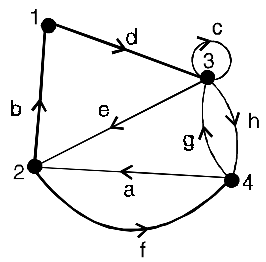


Fig. 8.23. Euler digraph.

Theorem 8.16. *In a connected, balanced digraph G of n vertices and m edges, let W =* (e_1, e_2, ..., e_m) *be an Euler line which starts and ends at a vertex v (that is, v is the initial vertex of* e_1 *and the terminal vertex of* e_m*). Among the m edges in W there are n – 1 edges that 'enter' each of n – 1 vertices, other than v, for the first time. The subdigraph g of these n – 1 directed edges together with the n vertices is a spanning arborescence of G, rooted at vertex v.*

Proof. In the subgraph g, vertex v is of in-degree zero, and every other vertex is of in-degree one ; for g includes exctly one edge going to each of the $n - 1$ vertices, and no edge going to v.

Moreover, the way g is defined in W, g is connected and contains $n - 1$ directed edges.

Therefore, g is a spanning arborescence in G and is rooted at v.

Theorem. 8.17. *In an arborescence there is a directed path from the root R to every other vertex. Conversely, a circuitless digraph G is an arborescence if there is a vertex v in G such that every other vertex is accessible from v, and v is not accessible from any other vertex.*

Proof. In an arborescence consider a directed path P starting from the root R and continuing as far as possible. P can end only at a pendant vertex, otherwise we get a vertex whose in-degree is two or more, a contradiction.

Since an arborescence is connected, every vertex lies on some directed path from the root R to each of the pendant vertices.

Conversely, since every vertex in G is accessible from v, and G has no circuit, G is a tree. Moreover, since v is not accessible from any other vertex, $d^{-}(v) = 0$.

Every other vertex is accessible from v and therefore the in-degree of each of these vertices must be at least one.

The in-degree cannot be greater than one because there are only $n - 1$ edges in G (n being the number of vertices in G.)

Theorem 8.18. *Let G be an Euler digraph and T be a spanning in-tree in G, rooted at a vertex R. Let e_1 be an edge in G incident out of the vertex R. Then a directed walk W = $(e_1, e_2, ..., e_m)$ is a directed Euler line, if it is constructed as follows :*

(*i*) *No edge is included in W more than once.*

(*ii*) *In exiting a vertex the one edge belonging to T is not used until all other outgoing edges have been traversed.*

(*iii*) *The walk is terminated only when a vertex is reached from which there is no edge left on which to exit.*

Proof. The walk W must terminate at R, because all vertices must have been entered as often as they have been left(because G is balanced).

Now suppose there is an edge a in G that has not been included in W.

Let v be the terminal vertex of a. Since G is balanced v must also be the initial vertex of some edge b not included in W. Edge b going out of vertex v must be in T according to rule (*i*). Thus omitted edge leads to another omitted edge c in T, and so on.

Ultimately, we arrive at R, and find an outgoing edge there not included in W. This contradicts rule (*iii*).

Theorem 8.19. *If A(G) is the incidence matrix of a connected digraph of n vertices, the rank of A(G) = n – 1.*

Theorem 8.20. *The $(i, j)^{th}$ entry in X^r equals the number of different directed edge sequences of r edges from the i^{th} vertex to the j^{th}.*

Proof. (By induction)

The theorem is trivially true for $r = 1$.

As the inductive hypothesis, assume that the theorem holds for X^{r-1}. The $(i, j)^{th}$ entry in

$$X^r\ (= X^{r-1}.\ X) = \sum_{k=1}^{n} [(i, k)^{th} \text{ entry in } X^{r-1}]\ .\ x_{kj}$$

$$= \sum_{k=1}^{n} \begin{array}{l}(\text{number of all directed edge sequences} \\ \text{of length } r - 1 \text{ from vertex } i \text{ to } k).x_{kj},\end{array} \qquad ...(1)$$

according to the induction hypothesis. In (1), $x_{kj} = 1$ or 0 depending on whether or not there is a directed edge from k to j. Thus a term in the sum (1) is non zero if and only if there is a directed edge sequence of length r from i to j, whose last edge is from k to j.

If the term is non zero, its value equals the number of such edge sequences from i to j *via* k. This holds for every k, $1 \le k \le n$. Therefore (1) is equal to the number of all possible directed edge sequence from i to j.

Theorem 8.21. *Let B and A be respectively, the circuit matrix and incidence matrix of a self-loop-free digraph such that the columns in B and A are arranged using the same order of edges. Then*

$$A.B^T = B.A^T = 0.$$

Where superscript T denotes the transposed Matrix.

Proof. Consider the m^{th} row in B and the k^{th} row in A. If the circuit m does not include any edge incident on vertex k, the product of the two rows is clearly zero. If, on the other hand, vertex k is in circuit m, there are exactly two edges (say x and y) incident on k that are also in circuit m.

This situation can occur in only four different ways, as shown in Fig. 8.24 below.

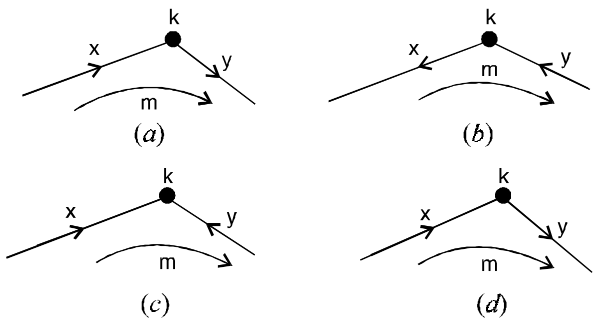


Fig. 8.24 : Vertex k in circuit m.

The possible entries in row k of A and row m of B in column positions x and y are tabulated for each of these four cases.

Case	*Row k*		*Row m*		*Dot product*
	column x	*column y*	*column x*	*column y*	*Row k . Row m*
(*i*)	– 1	1	1	1	0
(*ii*)	1	– 1	– 1	– 1	0
(*iii*)	– 1	– 1	1	– 1	0
(*iv*)	1	1	– 1	1	0

In each case, the dot product is zero. Therefore, the theorem.

Theorem 8.22. *The i, j entry* $a_{ij}^{(n)}$ *of* A^n *is the number of walks of length n from* v_i *to* v_j.

Corollary (1) The entries of the reachability and distance matrices can be obtained from the powers of A as follows :

(*i*) for all i, $r_{ii} = 1$ and $d_{ii} = 0$

(*ii*) $r_{ij} = 1$ if and only if for some n, $a_{ij}^{(n)} > 0$

(*iii*) $d(v_i, v_j)$ is the least n(if any) such that $a_{ij}^{(n)} > 0$, and is ∞ otherwise.

Corollary (2) Let v_i be a point of a digraph D. The strong component of D containing v_i is determined by the entries of 1 in the ith row (or column) of the matrix $R \times R^T$.

Theorem 8.23. *The value of the cofactor of any entry in the j^{th} column of M_{id} is the number of spanning out-trees with v_j as source.*

Corollary. In an Eulerian digraph, the number of eulerian trails is C. $\prod_{i=1}^{P} (d_i - 1)!$

Where $d_i = id(v_i)$ and c is the common value of all the cofactors of M_{od}.

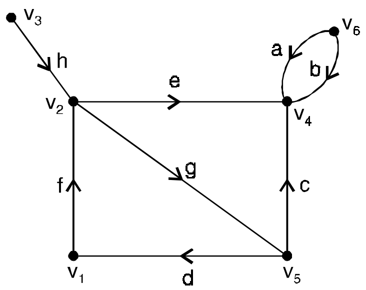
Theorem 8.24. *For any labeled digraph D, the value of the cofactor of any entry in the i^{th} row of M_{od} is the number of spanning in-trees with v_i as sink.*

Theorem 8.25. *The determinant of every square submatrix of A, the incident matrix of a digraph is 1, – 1 or 0.*

Proof. The theorem can be proved directly by expanding the determinant of a square submatrix of A.

Consider a k by k submatrix M of A.

If M has any column or row consisting of all zeros det M is clearly zero. Also det M = 0 if every column of M contains the two non zero entries, a^1 and a^{-1}.



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

(*a*) Digraph **(*b*) Incidence matrix**

Fig. 8.25 : Digraph and its incidence matrix.

Now if det M ≠ 0 (*i.e.*, M is non singular), then the sum of entries in each column of M cannot be zero.

Therefore, M must have a column in which there is a single non zero element that either + 1 or – 1.

Let this single element be in the $(i, j)^{th}$ position in M.

Thus det M = ± 1. det M_{ij}, where M_{ij} is the sub matrix of M with its i^{th} row and j^{th} column deleted.

The $(k - 1)$ by $(k - 1)$ submatrix M_{ij} is also non singular. Therefore it too must have atleast one column with a single non zero entry, say, in the $(p, q)^{th}$ position.

Expanding det M_{ij} about this element in the $(p, q)^{th}$ position.

det M_{ij} = ± [determinant of a non sigular $(k - 2)$ by $(k - 2)$ submatrix of M]

Repeated application of this procedure yileds det M = ± 1.

Hence the theorem.

Theorem 8.26. *Let A_f be the reduced incidence matrix of a connected digraph. Then the number of spanning trees in the graph equals the value of det $(A_f \, . \, T_f^T)$.*

Proof. According to the Binet-Cauchy theorem

det $(A_f . A_f^T)$ = sum of the products of all corresponding majors of A_f and A_f^T.

Every major of A_f or A_f^T is zero unless it corresponds to a spanning tree, in which case its value is 1 or – 1. Since both majors of A_f and A_f^T have the same value + 1 or – 1, the product is + 1 for each spanning tree.

Theorem 8.27. *Let k(G) be the Kirchhoff matrix of a simple digraph G. Then the value of the (p, q) cofactor of k(G) is equal to the number of arborescences in G rooted at the vertex v_q.*

Proof. The determinant of a square matrix is a linear function of its columns. Specifically, if p is a square matrix consisting of n column vectors, each of dimension n ; that is ;

$$P = [P_1, P_2, \ldots (P_i + P_i'), \ldots P_n]$$

then det P $\qquad$ $\det = [P_1, P_2, \ldots, P_i, \ldots, P_n] + \det [P_1, P_2, \ldots, P_i' \ldots P_n]$ ...(1)

In graph G suppose that vertex v_j has in-degree of d_j. The j^{th} column of k(G) can be regarded as the sum of d_j different columns, each corresponding to a graph in which v_j has in-degree one. And then (1) can be repeatedly applied. After this, splitting of columns can be carried out for each j, $j \neq q$, and det k_{qq}(G) can be expressed as a sum of determinants of subgraphs ; that is ; det $k_{qq}(G) = \sum_{g} \det k_{qq}(g)$,

...(2)

Where g is a subgraph of G, with the following properties :

(*i*) Every vertex in g has an in-degree of exactly one, except v_q.

(*ii*) g has n – 1 vertices, and hence n – 1 edges

det $k_{qq}(g)$ = 1, if and only if g is an orborescence rooted at q,

= 0, otherwise.

Thus the summation in (2) carried over all g's equals the number of arborescences rooted at v_q.

Theorem 8.28. *In an Euler digraph the number of Euler lines is*

$$\sigma \, . \prod_{i=1}^{n} [d^-(v_i) - 1]!$$

Theorem 8.29. *A simple digraph G of n vertices and n – 1 directed edges in an arborescence rooted at v_1 if and only if the (1, 1) cofactor of k(G) is equal to 1.*

Proof. Let G be an arborescence with n vertices and rooted at vertex v_1. Relabel the vertices as $v_1, v_2, \ldots v_n$ such that vertices along every directed path from the root v_1 have increasing indices.

Permute the rows and columns of k(G) to conform with this relabeling.

Since the in-degree of v_1 equals zero, the first column contains only zeros. Other entries in k(G) are

$k_{ij} = 0, \; i > j,$

$k_{ij} = -x_{ij}, i < j,$

$k_{ij} = 1, \; i > 1.$

Then the k matrix of an arborescence rooted at v_1 is of the form

$$k(G) = \begin{bmatrix} 0 & -x_{12} & -x_{13} & -x_{14} & \ldots\ldots & -x_{1n} \\ 0 & 1 & -x_{23} & -x_{24} & \ldots\ldots & -x_{2n} \\ 0 & 0 & 1 & -x_{34} & \ldots\ldots & -x_{3n} \\ 0 & 0 & 0 & 1 & \ldots\ldots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \ldots\ldots & 1 \end{bmatrix}$$

Clearly, the cofactor of the (1, 1) entry is 1.

That is, det $k_{11} = 1$.

Conversely, let G be a simple digraph of n vertices and $n - 1$ edges and let the (1, 1) cofactor of its k matrix be equal to 1 : that is ; det $k_{11} = 1$.

Since det $k_{11} \neq 0$, every column in k_{11} has at least one non zero entry.

Therefore $d^-(v_i) \geq 1$, for $i = 2, 3, \ldots.., n$.

There are only $n - 1$ edges to go around.

Therefore, $d^-(v_i) = 1$, for $i = 2, 3, \ldots.., n$ and $d^-(v_1) = 0$.

Now since no vertex in G has an in-degree of more than one, if G can have any circuit at all, it has to be a directed circuit.

Suppose that such a directed circuit exists ; which passes through vertices $v_{i_1}, v_{i_2}, \ldots\ldots, v_{i_r}$.

Then the sum of the columns $i_1, i_2, \ldots.. i_r$ in k_{11} is zero.

Thus these r columns in k_{11} are linearly dependent.

Hence det $k_{11} = 0$, a contradiction.

Therefore, G has no circuits.

If G has $n - 1$ edges and no circuits, it must be a tree. Since in this tree $d^-(v_1) = 0$ and $d^-(v_i) = 1$ for $i = 2, 3, \ldots., n$.

G must be an arborescence rooted at vertex v_1.

The above arguments are valid for an arborescence at any vertex v_q. Any reordering of the vertices in G corresponds to identical permutations of rows and columns in k(G). Such permutations do not alter the value or sign of the determinant.

Problem 8.4. *Verify that the following two digraphs are isomorphic.*

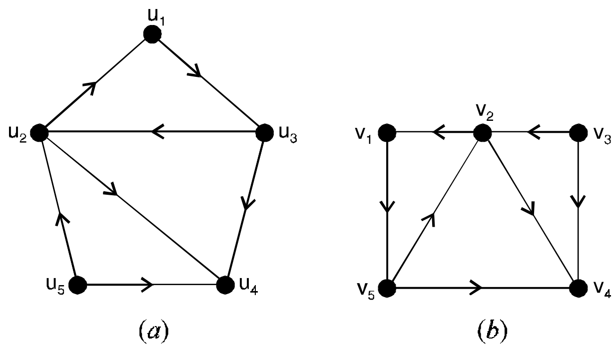


Fig. 8.26.

Solution. Let us consider the following one-to-one correspondence between the directed edgs in the two digraphs :

$$(u_2, u_1) \leftrightarrow (v_2, v_1), \qquad (u_5, u_2) \leftrightarrow (v_3, v_2)$$
$$(u_2, u_4) \leftrightarrow (v_2, v_4), \qquad (u_3, u_2) \leftrightarrow (v_5, v_2)$$
$$(u_3, u_4) \leftrightarrow (v_5, v_4), \qquad (u_5, u_4) \leftrightarrow (v_3, v_4)$$
$$(u_1, u_3) \leftrightarrow (v_1, v_5).$$

These yield the following one-to-one correspondence between the vertices in the two digraphs :

$$u_1 \leftrightarrow v_1,\ u_2 \leftrightarrow v_2,\ u_3 \leftrightarrow v_5,\ u_4 \leftrightarrow v_4,\ u_5 \leftrightarrow v_3.$$

The above mentioned one-to-one correspondences between the vertices and the directed edges establish the isomorphism between the given digraphs.

Problem 8.5. *Prove that a complete symmetric digraph of n vertices contains n(n – 1) edges and a complete asymmetric digraph of n vertices contains* $\dfrac{n(n-1)}{2}$ *edges.*

Solution. In a complete asymmetric digraph, there is exactly one edge between every pair of vertices.

Therefore, the number of edges in such a digraph is precisely equal to the number of pairs of vertices. The number of pairs of vertices that can be chosen from n vertices is ${}^nC_2 = \dfrac{1}{2}n(n-1)$.

Thus, a complete asymmetric digraph with n vertices has exactly $\dfrac{1}{2}n(n-1)$ edges.

In a complete symmetric digraph there exist two edges with opposite directions between every pair of vertices.

Therefore, the number of edges in such a digraph with n vertices is $2 \times \dfrac{1}{2}n(n-1) = n(n-1)$.

Problem 8.6. *Let D be a connected simple digraph with n vertices and m edges. Prove that*

$$n - 1 \le m \le n(n - 1).$$

Solution. Since D is connected, its underlying graph G is connected. Therefore, $m \ge n - 1$.

In a simple digraph, there exists at most two edges in opposite directions between every pair of vertices.

Therefore, the number of edges in such a digraph cannot exceed $2 \times {}^nC_2 = n(n - 1)$.

i.e., $m \le n(n - 1)$. Thus $n - 1 \le m \le n(n - 1)$.

Problem 8.7. *Find the sequence of vertices and edges of the longest walk in the digraph shown in Fig. 8.27 below :*

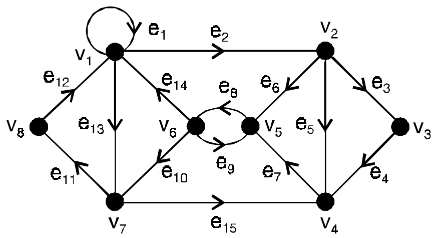


Fig. 8.27.

Solution. We check that in the given digraph, for each vertex, the in-degree is equal to the out-degree.

Therefore, the digraph is an Euler digraph. The longest walk in the digraph is a directed Euler line, a directed walk which includes all the edges of the digraph.

The digraph reveals that the directed Euler line is shown below :

$$v_1\ e_1\ v_1\ e_{13}\ v_7\ e_{11}\ v_8\ e_{12}\ v_1\ e_2\ v_2\ e_3\ v_3\ e_4\ v_4\ e_5\ v_2\ e_6\ v_5\ e_8\ v_6\ e_{10}\ v_7\ e_{15}\ v_4\ e_7\ v_5\ e_9\ v_6\ e_{14}\ v_1.$$

This is the required sequence.

Problem 8.8. *Show that the digraph shown in Fig. 8.28 below, is an Euler digraph. Indicate a directed Euler line in it.*

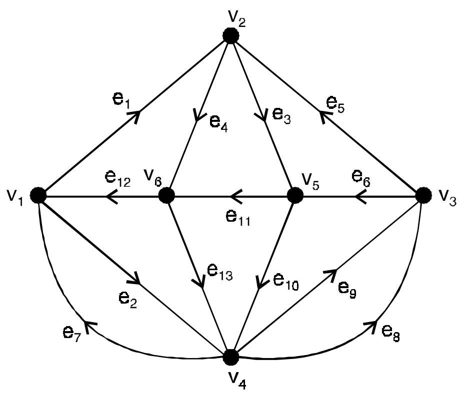


Fig. 8.28.

Solution. By examining the given digraph, we find that

$$d^-(v_1) = 2 = d^+(v_1),\ d^-(v_2) = 2 = d^+(v_2)$$
$$d^-(v_3) = 2 = d^+(v_3),\ d^-(v_4) = 3 = d^+(v_3)$$
$$d^-(v_5) = 2 = d^+(v_5),\ d^-(v_6) = 2 = d^+(v_6).$$

Thus, for every vertex the in-degree is equal to the out-degree.

Therefore the digraph is an Euler digraph.

By starting at v_1, we can obtain the following closed directed walk that includes all the thirteen edges :

$v_1\ e_1\ v_2\ e_4\ v_6\ e_{12}\ v_1\ e_2\ v_4\ e_8\ v_3\ e_6\ v_5\ e_{11}\ v_6\ e_{13}\ v_4\ e_9\ v_3\ e_5\ v_2\ e_3\ v_5\ e_3\ v_5\ e_{10}\ v_4\ e_7\ v_1$

This is a directed Euler line in the given digraph.

Problem 8.9. *Prove that a connected digraph D that does not contain a closed directed walk must have a source and a sink.*

Solution. Consider a directed walk q in D, which contains a maximum number of vertices.

Let u be the initial vertex of q and v be the terminal vertex. Suppose v is not a sink. Then there must be an edge that begins at v. Since D has no closed walk, this edge cannot end at u.

Hence it must end at some vertex v'.

Consequently, there is created a directed walk q' that contains all vertices of q and v'.

This contradicts the maximality of q. Hence v has to be a sink. Similarly, u has to be a source.

Thus, D contains at least one sink and at least one source.

Problem 8.10. *Consider the digraph D shown in Fig. 8.29 below. In this digraph find*

(*i*) *a directed walk of length 8. Is this walk a directed path ?*
(*ii*) *a directed trial of length 10*
(*iii*) *a directed trial of longest length*
(*iv*) *a directed path of longest length.*

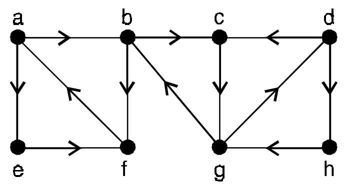


Fig. 8.29.

Solution. (*i*) *a e f a b c g d h* is a directed walk of length 8. This is not a directed path, because the vertex a appear twice.

(*ii*) *a e f a b c g d h g b* is a directed trial (walk) of length 10.

(*iii*) *a e f a b c g d h g b f* is a directed trial of length 11. This is the longest possible directed trial (walk).

(*iv*) *a b c g d h* is a directed path of length 5. This is the longest possible directed path.

Problem 8.11. *Show that the digraph given below is strongly connected.*

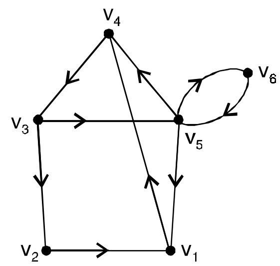


Fig. 8.30.

Solution. A digraph D is strongly connected if every vertex of D is rechable from every other vertex of D.

Let us verify wheather this condition holds for the given digraph.

The following table indicates the directed paths from each of the vertices v_1, v_2, v_3, v_4, v_5, v_6 to every other vertex. The existence of these directed paths proves that the given digraph is strongly connected.

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	—	$v_1v_4v_3v_1$	$v_1v_4v_3$	v_1v_4	$v_1v_4v_3v_5$	$v_1v_4v_3v_5v_6$
v_2	v_2v_1	—	$v_2v_1v_4v_3$	$v_2v_1v_4$	$v_2v_1v_4v_3v_5$	$v_2v_1v_4v_3v_5$
v_3	$v_3v_2v_1$	v_3v_2	—	$v_3v_5v_4$	v_3v_5	$v_3v_5v_6$
v_4	$v_4v_3v_2v_1$	$v_4v_3v_2$	v_4v_3	—	$v_4v_3v_5$	$v_4v_3v_5v_6$
v_5	v_5v_1	$v_5v_4v_3v_2$	$v_5v_4v_3$	v_5v_4	—	v_5v_6
v_6	$v_6v_5v_1$	$v_6v_5v_4v_3v_2$	$v_6v_5v_4v_3$	$v_6v_5v_4$	v_6v_5	—

8.26. NULLITY OF A MATRIX

If Q is an n by n matrix then QX = 0 has a non trial solution $X \neq 0$ if and only if Q is singular, that is ; det Q = 0. The set of all vectors X that satisfy QX = 0 forms a vector space called the null space of matrix Q. The rank of the null space is called the nullity of Q.

Rank of Q + nullity of Q = n

When Q is not square but a k by n matrix, $k < n$.

Theorem 8.30. ***(Binet-Cauchy Theorem)***

If Q and R are k by m and m × k matrices respectively with k < m then the determinant of the product det (QR) = the sum of the products of corresponding major determinants of Q and R.

Proof. To evaluate det (QR), let us devise and multiply two $(m + k)$ by $(m + k)$ partitioned matrices

$$\begin{bmatrix} I_k & Q \\ O & I_m \end{bmatrix} \cdot \begin{bmatrix} Q & O \\ -I_m & R \end{bmatrix} = \begin{bmatrix} O & QR \\ -I_m & R \end{bmatrix}$$

where I_m and I_k are identity matrices of order m and k respectively.

Therefore $\det \begin{bmatrix} Q & O \\ -I_m & R \end{bmatrix} = \det \begin{bmatrix} O & QR \\ -I_m & R \end{bmatrix}$

that is, $$\det (QR) = \det \begin{bmatrix} Q & O \\ -I_m & R \end{bmatrix} \qquad ...(1)$$

Let us now apply Cauchy's expansion method to the right-hand side of equation (1), and observe that the only non-zero minors of any order in matrix $-I_m$ are its principal minors of that order. We thus find that the Cauchy expansion consists of these minors of order $m - k$ multiplied by their cofactors of order k in Q and R together.

Theorem 8.31. ***(Sylvester's Law)***

If Q is a k by n matrix and R is an n by P matrix then the nullity of the product cannot exceed the sum of the nullities of the factors, that is ;

$$\text{nullity of } QR \leq \text{nullity of } Q + \text{nullity of } R \qquad ...(1)$$

Proof. Since every vector x that satisfies RX = 0 must certainly satisfy $QRx = 0$

We have nullity of QR $\geq$ nullity of R ≥ 0 ...(2)

Let s be the nullity of matrix R. Then there exists a set of s linearly independent vectors $\{x_1, x_2, ... x_s\}$ forming a basis of the null space of R.

Thus $Rx_i = 0$ for $i = 1, 2, ..., s$ ...(3)

Now let $s + t$ be the nullity of matrix QR. Then there must exist a set of t linearly independent vectors

$$\{x_{s+1}, x_{s+2}, ... x_{s+t}\} \text{ such that the set } \{x_1, x_2, ... x_s, x_{s+1}, x_{s+2}, x_{s+t}\}$$

forms a basis for the null space of matrix QR.

Thus $QRx_i = 0$, for $i = 1, 2, ... s, s + 1, s + 2, ... s + t$ ...(4)

In otherwords, of the $s + t$ vectors x_i forming a basis of the null space of QR, the first s vectors are sent to zero by matrix R and the remaining non-zero Rx_i's ($i = s + 1, s + 2, ... s + t$) are sent to zero by matrix Q.

Vectors $Rx_{s+1}, Rx_{s+2}, ... Rx_{s+t}$ are linearly independent ; for if

$$0 = a_1 Rx_{s+1} + a_2 Rx_{s+2} + ... + a_t Rx_{s+t}$$
$$= R(a_1 x_{s+1} + a_2 x_{s+2} + ... + a_t x_{s+t})$$

then vector $(a_1 x_{s+1} + a_2 x_{s+2} + ... + a_t x_{s+t})$ must be the null space of R, which is possible only if

$$a_1 = a_2 = ... = a_t = 0.$$

Thus we have found that there are at least t linearly independent vectors which are sent to zero by matrix Q and therefore nullity of Q $\geq t$.

But since $t = (s + t) - s$

$= \text{nullity of QR} - \text{nullity of R}$, equation (1) follows.

Substituting equation, rank of Q + nullity of Q = n into equation (1), we find that

$$\text{rank of QR} \geq \text{rank of Q} + \text{rank of R} - n \quad \text{...(5)}$$

Furthermore, in equation (5) if the matrix product QR is zero, then

$$\text{rank of Q} + \text{rank of R} \leq n.$$

Theorem 8.32. *If G is a (p, q) graph whose points have degrees d_i, then L(G) has q points and q_L lines where* $q_L = -q + \frac{1}{2}\sum d_i^2$.

Proof. By the definition of line graph, L(G) has q points. The d_i lines incident with a point v_i contribute $\binom{d_i}{2}$ to q_L, so

$$q_L = \sum \binom{d_i}{2} = \frac{1}{2}\sum d_i(d_i - 1)$$

$$= \frac{1}{2}\sum d_i^2 - \frac{1}{2}\sum d_i$$

$$= \frac{1}{2}\sum d_i^2 - q .$$

Theorem 8.33. *Unless m = n = 4, a graph G is the line graph of $k_{m,n}$ if and only if*

(*i*) *G has mn points*

(*ii*) *G is regular of degree m + n – 2.*

(*iii*) *Every two non adjacent points are mutually adjacent to exactly two points*

(*iv*) *Among the adjacent pairs of points, exactly* $n\binom{m}{2}$ *pairs are mutually adjacent to exactly m – 2 points, and other* $m\binom{n}{2}$ *pairs to n – 2 points.*

Theorem 8.34. *Unless P = 8, a graph G is the line graph of k_P if and only if*

(*i*) *G has* $\binom{P}{2}$ *points,*

(*ii*) *G is regular of degree 2(P – 2),*

(*iii*) *Every two non adjacent points are mutually adjacent to exactly four points,*

(*iv*) *Every two adjacent points are mutually adjacent to exactly P – 2 points.*

Theorem 8.35. *The total graph T(G) is isomorphic to the square of the sub-division graph S(G).*

Corollary (1). If v is a point of G then the degree of point v in T(G) is 2 deg v. If $x = uv$ is a line of G then the degree of point x in T(G) is deg u + deg v.

Corollary (2). If G is a (p, q) graph whose points have degrees d_i, then the total graph T(G) has $P_T = p + q$ points and $q_T = 2q + \frac{1}{2}\sum d_i^2$ lines.

Theorem 8.36. *If G is a non trivial connected graph with P points which is not a path, then $L^n(G)$ is hamiltonian for all $n \geq P - 3$.*

Theorem 8.37. *For $n > 1$, we always have $r_1(2, n) = 3$. For all other m and n, $r_1(m, n) = (m - 1)(n - 1) + 2$.*

Theorem 8.38. *A graph G is eulerian if and only if $L_3(G)$ is hamiltonian.*

Theorem 8.39. *Let G and G′ be connected graphs with isomorphic line graphs. Then G and G′ are isomorphic unless one is k_3 and the other is $k_{1,3}$.*

Proof. First note that among the connected graphs with up to four points, the only two different ones with isomorphic line graphs are k_3 and $k_{1,3}$.

Note further that if ϕ is an isomorphism of G and G′ then there is a derived isomorphism ϕ_1 of L(G) onto L(G′).

The theorem will be demonstrated when the following stronger result is proved.

If G and G′ have more than four points then any isomorphism ϕ_1 of L(G) and L(G′) is derived from exactly one isomorphism of G to G′.

We first show that ϕ_1 is derived from at most one isomorphism.

Assume there are two such, ϕ and ψ. We will prove that for any point v of G, $\phi(v) = \psi(v)$.

There must exist two lines $x = uv$ and $y = uw$ or vw.

If $y = vw$ then the points $\phi(v)$ and $\psi(v)$ are on both lines $\phi_1(x)$ and $\phi_1(y)$, so that since only one point is on both these lines, $\phi(v) = \psi(v)$.

By the same argument, when $y = uw$, $\phi(u) = \psi(u)$ so taht since the line $\phi_1(x)$ contains the two points $\phi(v)$ and $\phi(u) = \psi(u)$, we again have $\phi(v) = \psi(v)$.

Therefore ϕ_1 is derived from at most one isomorphism of G to G′.

We now show the existence of an isomorphism ϕ from which ϕ_1 is derived.

The first step is to show that the lines $x_1 = uv_1$, $x_2 = uv_2$, and $x_3 = uv_3$ of a $k_{1,3}$ subgraph of of G must go to the lines of a $k_{1,3}$ subgraph of G′ under ϕ.

Let y be another line adjacent with at least one of the x_i, which is adjacent with only one or all three. Such a line y must exist for any graph with $P \geq 5$ and the theorem is trivial for $P < 5$.

If the three lines $\phi_1(x_i)$ form a triangle instead of $k_{1,3}$ the $\phi_1(y)$ must be adjacent with precisely two of the three.

Therefore, every $k_{1,3}$ must go to a $k_{1,3}$.

Let $s(v)$ denote the set of lines at v. We now show that to each v in G, there is exactly one v' in G′ such that S(v) goes to S(v') under ϕ_1.

If deg $v \geq 2$, let y_1 and y_2 be lines at v and let v' be the common point of $\phi_1(y)$ and $\phi_1(y_2)$.

Then for each line x at v, v' is incident with $\phi_1(x)$ and for each line x' and v', v is incident with $\phi_1^{-1}(x')$.

If deg $v = 1$, let $x = uv$ be the line at v.

Then deg $u \geq 2$ and hence $s(u)$ goes to $s(u')$ and $\phi_1(x) = u'v'$.

Since for every line x' at v', the lines $\phi_1^{-1}(x')$ and x must have a common point, u is on $\phi_1^{-1}(x')$ and u' is on x'.

That is, $x' = \phi_1(x)$ and deg $v' = 1$. The mapping ϕ is therefore one-to-one from v to v' since $s(u) = s(v)$ only when $u = v$.

Now given v' in V′, there is an incident line x'.

Denote $\phi_1^{-1}(x')$ by uv. The either $\phi(u) = v'$ or $\phi(v) = v'$ so ϕ is onto.

Finally, we note that for each line $x = uv$ in G, $\phi_1(x) = \phi(u)\phi(v)$ and for each line $x' = u'v'$ in G′, $\phi_1^{-1}(x') = \phi^{-1}(u')\ \phi^{-1}(v')$, so that ϕ is an isomorphism from which ϕ_1 is derived.

This complete the proof.

Theorem 8.40. *A connected graph is isomorphic to its line graph if and only if it is a cycle.*

Theorem 8.41. *A sufficient condition for $L_2(G)$ to be hamiltonian is that G be hamiltonian and a necessary condition is that L(G) be hamiltonian.*

Theorem 8.42. *If G is eulerian then L(G) is both eulerian and hamiltonian. If G is hamiltonian then L(G) is hamiltonian.*

Theorem 8.43. *A graph is the line graph of a tree if and only if it is a connected block graph in which each cut point is on exactly two blocks.*

Proof. Suppose G = L(T), T some tree.

Then G is also B(T) since the lines and blocks of a tree coincide.

Each cut point x of G corresponds to a bridge uv to T, and is on exactly those two blocks of G which correspond to the stars at u and v. This proves the necessity of the condition.

Sufficient part

Let G be a block graph is which each cutpoint is on exactly two blocks.

Since each block of a block graph is complete, there exists a graph H such that L(H) = G.

If G = K_3, we can take H = $K_{1,3}$.

If G is any other block graph, then we show that H must be a tree.

Assume that H is not a tree so that it contains a cycle. If H is itself a cycle then L(H) = H, but the only cycle which is a block graph is K_3, a case not under consideration.

Hence H must properly contain a cycle, there by implying that H has a cycle Z and a line x adjacent to two lines of Z, but not adjacent to some line y of Z. The points x and y of L(H) lie on a cycle of L(H) and they are not adjacent.

This contradicts that L(H) is a block graph.

Hence H is a tree and the theorem is proved.

Theorem 8.44. *The following statements are equivalent :*

(*i*) *G is a line graph*

(*ii*) *The lines of G can be partitioned into complete subgraphs in such a way that no point lies in more than two of the subgraphs.*

Proof. (*i*) implies (*ii*)

Let G be the line graph of H. Withoutloss of generality we assume that H has no isolated points. Then the lines in the star at each point of H induce a complete subgraph of G, and every line of G lies in exactly one such subgraph.

Since each line of H belongs to the stars of exactly to points of H, no points of G is in more than two of these complete subgraphs.

(*ii*) implies (*i*)

Given a decomposition of the lines of a graph G into complete subgraphs $S_1, S_2, ..., S_n$ satisfaying (*ii*), we indicate the construction of a graph H whose line graph is G.

The points of H correspond to the set S of subgraphs of the decomposition together with the set U of points of G belonging to only one of the subgraphs S_i.

Thus $S \cup U$ is the set of points of H and two of these points are adjacent whenever they have a non empty intersection ; that is ; H is the intersection graph Ω $(S \cup U)$.

Theorem 8.45. *Every tournament has a spanning path.*

Proof. The proof is by induction on the number of points.

Every tournament with 2, 3, or 4 points has a spanning path, by inspection.

Assume the result is true for all tournament with n points, and consider a tournament T with $n + 1$ points.

Let v_0 be any point of T. Then $T - v_0$ is a tournament with n points, so it has a spanning path P, say $v_1 v_2 .... v_n$.

Either arc $v_0 v_1$ or arc $v_1 v_0$ is in T. If $v_0 v_1$ is in T, then $v_0 v_1 v_2 ..... v_n$ is a spanning path of T.

If $v_1 v_0$ is in T, let v_i be the first point of P for which the arc $v_0 v_i$ is in T, if any.

Then $v_{i-1} v_0$ is in T, so that $v_1 v_2 ..... v_{i-1} v_0 v_i ..... v_n$ is a spanning path.

If no such point v_i exists, then $v_1 v_2 ...... v_n v_0$ is a spanning path. In any case, we have shown that T has a spanning path, completing the proof.

Theorem 8.46. *The distance from a point with maximum score to any other point is 1 or 2.*

Theorem 8.47. *The number of transitive triples in tournament with score sequence $(S_1, S_2, ...,$ $S_P)$ is* $\sum \frac{S_i(S_i - 1)}{2}$.

Corollary. The maximum number of cycle triples among all tournaments with P points is

$$t(P, 3) = \begin{cases} \dfrac{P^3 - P}{24} & \text{if } P \text{ is odd,} \\ \dfrac{P^3 - 4P}{24} & \text{if } P \text{ is even.} \end{cases}$$

Theorem 8.48. *Every strong tournament with P points has a cycle of length n, for n = 3, 4, ... P.*

Proof. This proof is also by induction, but on the length of cycles.

If a tournament T is strong, then it must have a cycle triple.

Assume that T has a cycle Z = v_1v_2 ...... v_nv_1 of length n < P.

We will show that it has a cycle of length n + 1.

There are two cases : either there is a point u not in Z both adjacent to and adjacent from points of Z, or there is no such point.

Case (*i*) Assume there is a point u not in Z and points v and w in Z such that arcs uv and wu are in T. Without loss of generality, we assume that arc v_1u is in T.

Let v_i be the first point, going around Z from v_1, for which arc uv_i is in T. Then v_{i-1} u is in T, and v_1v_2 ...... v_{i-1} uv_i ..... v_nv_1 is a cycle of length n + 1.

Case (*ii*) There is no such point u as in case (*i*). Hence, all points of T which are not in Z are partitioned into the two subsets U and W, where U is the set of all points adjacent to every point of Z and W is the set adjacent from every point of Z.

Clearly, these sets are disjoint, and neither set is empty since otherwise T would not be strong.

Furthermore, there are points u in U and w in W such that arc wu is in T.

Therefore uv_1v_2 ..... v_{n-1} wu is a cycle of length, n + 1 in T.

Hence, there is a cycle of length n + 1, completing the proof.

Corollary. A tournament is strong if and only if it has a spanning cycle.

Problem Set 8.1

1. Prove that every digraph in which idv, $odv \geq \frac{P}{2}$ for all points v is hamiltonian.
2. Prove that the line digraph L(D) of a weak digraph D is isomorphic to D if and only if D or D′ is functional.
3. Prove that the number of Eulerian trails of a digraph D equals the number of hamiltonian cycles of L(D).
4. Prove that, every orientation of an n-chromatic graph G contains a path of length n – 1.
5. Let D be a primitive digraph
   (*i*) If n is the smallest integer such that $A^n > 0$ then show that $n \leq (P-1)^2 + 1$.
   (*ii*) If n has the maximum possible value $(P-1)^2 + 1$, then show that there exists a permutation matrix P such that PAP^{-1} has the form $[a_{ij}]$ where $a_{ij} = 1$ whenever $j = i + 1$ and $a_{p,1} = 1$ but $a_{ij} = 0$ otherwise.
6. Let A be the adjacency matrix of the line digraph of a complete symmetric digraph then show that $A^2 + A$ has all entries 1.
7. Prove that, there exists a digraph with outdegree sequence (S_1, S_2, ... S_P) where $P - 1 \geq S_1 \geq S_2 \geq$ .... $\geq S_P$, and indegree sequence (t_1, t_2, ... t_n) where every $t_j \geq P - 1$ if and only if $\Sigma S_i = \Sigma t_i$ and for each integer k < P, $\sum_{i=1}^{k} S_i \leq \sum_{i=1}^{k} \min\{k-1, t_i\} + \sum_{i=k+1}^{P} \min\{k, t_i\}$.
8. Prove that the determinant of every square submatrix of A, the incidence matrix of a digraph 1, – 1 or 0.

9. Let B and A be respectively, the circuit matrix and incidence matrix of a self-loop free digraph such that the columns in B and A are arranged using the same order of edges. Then show that A . B^T = B . A^T = 0.

10. Prove that a simple digraph G of n vertices and $n - 1$ directed edges is an arborescence rooted at v_1 if and only if the (1, 1) cofactor of k(G) is equal to 1.

11. Let k(G) be the Kirchhoff matrix of a simple digraph G. Then prove that the value of the (q, q) cofactor of k(G) is equal to the number of arborescences in G rooted at the vertex v_q.

12. Prove that, every complete tournament has a directed Hamiltonian path.

13. Prove that, a digraph G is a cyclic if and only if its vertices can be ordered such that the adjacency matrix X is an upper or lower triangular matrix.

14. Prove that, every acyclic digraph G has at least one vertex with zero in-degree and at least one vertex with zero out-degree.

15. Prove that, digraph G is acyclic if and only if det (I – X) is not equal to zero. Where I is the identity matrix of the same size as X.

16. Prove that every edge in a digraph belongs either to a directed circuit or a directed cut-set.

17. Prove that in any digraph the sum of the in-degrees of all vertices is equal to the sum of their out-degrees, and this sum is equal to the number of edges in the digraph.

18. Prove that every Euler digraph (without isolated vertices) is strongly connected. Also show by constructing a counter example, that the converse is not true.

19. Prove that an n-vertex digraph is strongly connected if and only if the matrix M, defined by $M = x + x^2 + x^3 + \ldots\ldots + x^n$, has no zero entry, x is the adjacency matrix.

20. Prove that the number of directed Euler lines in GD(r) is $2^{2^{r-1} - r}$.

21. Prove that an acyclic digraph G of n vertices has a unique directed Hamiltonian path if and only if the number of non zero elements in R(G) is $\dfrac{n(n-1)}{2}$.

22. Prove that for every $n \geq 3$ these exists at least one acyclic complete tournament of n vertices.

23. Prove that a digraph G is acyclic if and only if every element on the principal diagonal of its reachability or accessibility matrix R(G) is zero.

24. If E | G | is the number of Euler lines in an n-vertex Euler digraph G, show that 2^{n-1} . E | G | is the number of Euler line in L(G).

25. Show that if R is the reachability matrix of a digraph G the value of the i^{th} entry in the principal diagonal R^2 gives the number of vertices included in the strongly connected fragment containing the i^{th} vertex.

26. Prove that any acyclic digraph G is an arborescence if and only if there is a vertex v in G such that every vertex is accessible from v.

27. Let $t(x)$ and $s(x)$ be the generating functions for tournaments and strong tournaments, respectively, then prove that $s(x) = \dfrac{t(x)}{1 + t(x)}$.

28. Prove that, A graph is isomorphic to the point-group of some tournament if and only if it has odd order.

29. Prove that, the maximum number of strong subtournaments with 4 points in any P point tournament is $t(\text{P}, 4) = \frac{1}{2}(\text{P} - 3)t(\text{P}, 3)$.

30. Prove that, the number of cycles of length 4 in any P point tournament is equal to the number of strong subtournament with 4 points.

31. Prove that the complement $\overline{\text{D}}$ and the converse D′ both have the same group as D.

32. Let R(G) be the reachability matrix of a digraph G and let the vertices in G be the ordered such that the sums of the rows in R(G) are non increasing ; that is ;

$$\sum_{j=1}^{n} r_{ij} \geq \sum_{j=1}^{n} r_{kj} \quad \text{for every } i < k.$$

Show with this ordering of vertices in R(G) that digraph G is acyclic if and only if R(G) is an upper triangular matrix.

33. Show that the following two digraphs are not isomorphic.

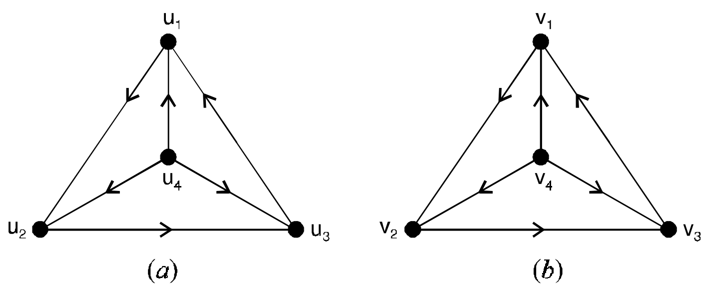


Fig. 8.31.

34. Let D be the digraph whose vertex is

V = $\{v_1, v_2, v_3, v_4, v_5\}$ and the edge set is

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

Write down a diagram of D and indicate the out degrees and in-degrees of vertices.

35. Show that the following digraphs are not isomorphic whereas their underlying graphs are isomorphic.

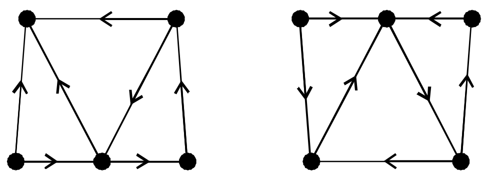


Fig. 8.32.

36. A directed walk in a digraph D which contains all vertices of D is called a spanning walk in D. Find the shortest spanning walk in the digraph in Fig. 8.33 below.

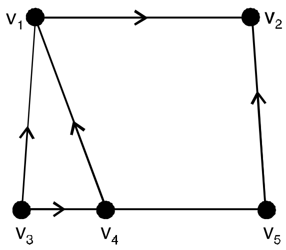


Fig. 8.33.

37. Prove that every edge in a digraph belongs to either a directed circuit or a directed cut-set.

38. Show that k_4 and $k_{2,\,3}$ are orientable graphs.

39. Find a directed Eulerian line and a spanning arborescence in the digraph shown in Fig. 8.34 below.

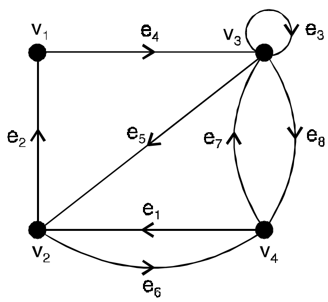


Fig. 8.34.

40. Find the adjacency matrix for the digraph shown in Fig. 8.35 below.

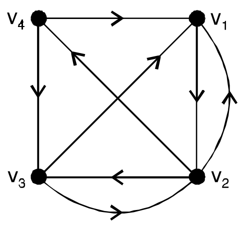


Fig. 8.35.

41. For the diagraph in Fig. 8.36 below, find all arborescence rooted at the vertex 4.

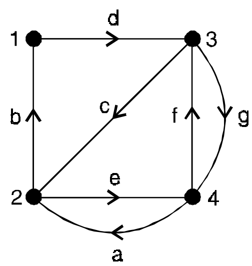


Fig. 8.36.

42. For the digraph in Fig. 8.37 below, find the incidence matrix :

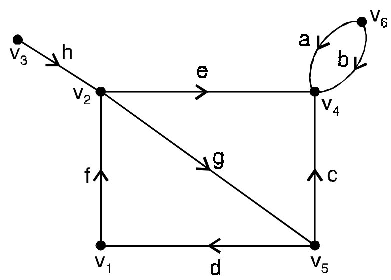


Fig. 8.37.

43. Show that, in a tournament, the sum of the squares of in-degrees of all vertices is equal to the sum of the sequences of out-degree.

44. Show that if a tournament has a directed circuit, it has a directed triangle.

45. Prove that a tournament cannot have more than one source and more than one sink.

46. Show that the following is an Euler digraph. Find a directed Euler line in it.

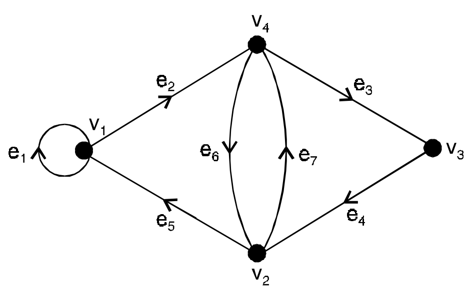


Fig. 8.38.

47. Prove that, a connected digraph D is an Euler digraph if and only if $d^-(v) = d^+(v)$ for every vertex v of D.

48. Prove that, every Euler digraph is strongly connected, but the converse is not necessarily true.

49. Prove that the condensation of a digraph D is strongly connected if and only if D is strongly connected.

50. For the digraph shown in Fig. 8.39 below, find a path of maximum length.

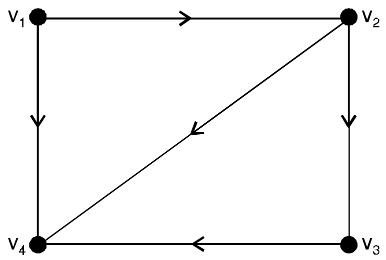


Fig. 8.39.

51. Find the fragments and condensation of the digraph shown in Fig. 8.40 below.

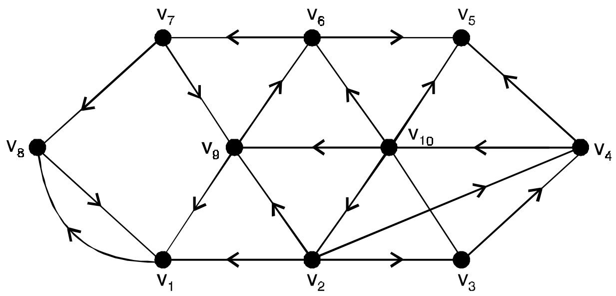
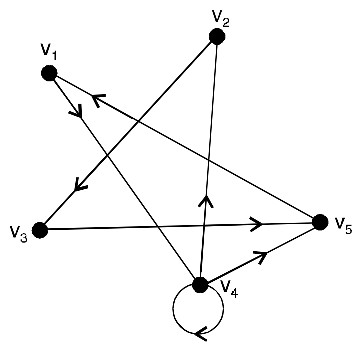


Fig. 8.40.

Answers 1.1

34.



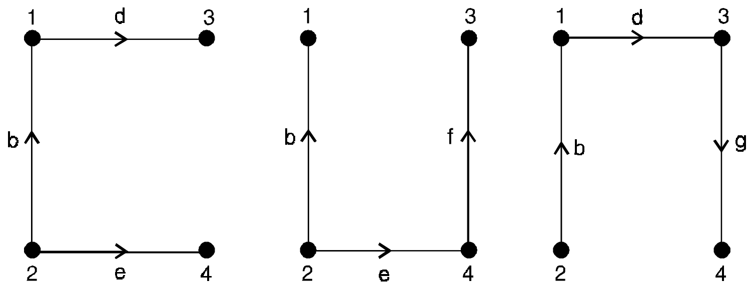
Vertices	v_1	v_2	v_3	v_4	v_5
d^+	1	1	1	3	1
d^-	1	1	1	2	2

39. Directed Euler line = $e_2e_4e_3e_5e_6e_7e_8e_1$

Spanning arborescence : $\{e_2, e_4, e_6\}$ rooted at v_2.

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

41.

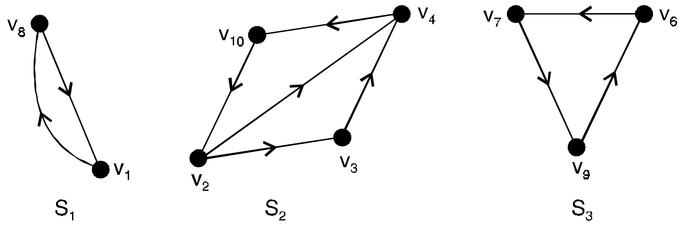


42. $\begin{bmatrix} 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

46. $v_1e_1v_1e_2v_4e_6v_2e_7v_4e_3v_3e_4v_2e_5v_1$.

47. $v_1v_2v_4v_3$.

51.



and