

# APPLICATIONS OF GRAPH THEORY

# **APPLICATIONS OF GRAPH THEORY**

*A PROJECT REPORT*

**Submitted**

**In partial fulfilment of the requirements for the award of degree**

**Master of Science**

**In**

**Mathematics**

**By**

**S.MANIKANTA  
(HT.NO:1683531014)**

**Under the esteemed guidance of**

**A.PADHMA**



**Department of Mathematics**

**GOVERNMENT COLLEGE (A), RAJAMAHENDRAVARAM**

**Affiliated by AKNU, Rajamahendravaram**

**Andhra Pradesh, India**

**2017-2018**

# **Certificate**

**This is to certify that the project entitled “APPLICATIONS OF GRAPH THEORY” is the bonafide work carried out by S.MANIKANTA during the academic year 2017-18 in partial fulfilment of the requirements for the award of the degree of master of science in dept. of mathematics , Government(A) College , Rajamahendravaram .**

**External Examiner**

**Signature of the Guide**

**signature of HOD**

*The matter embodied in this project work has not been submitted earlier for award of any degree or diploma to the best of my knowledge and belief. Certified that the above mentioned project has been duly carried out as per the norms of the college and statutes of the university*

# Acknowledgements

I wish to express my deep sense of thanks and gratitude to **A.PADHMA**, Department MATHEMATICS, Government (A) College, who guided me in intricacies of this project.

I would like to offer my special thanks to **DR.CH.SRINIVASULU**, Head of the Dept (MATHEMATICS), Government (A) College For his co-operation and support during project preparation.

I am particularly grateful to **Dr. Rapaka David Kumar Swamy**, principal, Government (A) College, for the guidelines given by him.

My special thanks to **Sri N. Kiran Kumar**, principal, V.SM College of RCPM for his valuable and constructive suggestions and assistance for development of this work

# CONTENTS

<b>Certificate</b>	<b>2</b>
<b>Acknowledgements</b>	<b>3</b>
<b>1. Graph theory</b>	<b>7-9</b>
1.1. History	
1.2. Definition	
<b>2. Terminology</b>	<b>9-12</b>
2.1. Loop	
2.2. Multigraph	
2.3. Directed and Un-directed graph	
2.4. Pseudo graph	
2.5. Simple graph	
2.6. Finite and Infinite graphs	
2.7. Degree of vertex	
2.8. Isolated and pendent vertices	
<b>3. Types of Graphs</b>	<b>12-15</b>
3.1. Null graph	
3.2. Complete graph	
3.3. Regular graph	
3.4. Cycles	
3.5. Wheels	
3.6. Platonic graph	
3.7. N-Cube	
<b>4. Tree and Forest</b>	<b>15-16</b>
4.1. Tree	
4.2. Forest	
4.3. Poly tree	
4.4. Types of trees	
<b>5. Graph isomorphism and Graph operations</b>	<b>17-23</b>
5.1. Graph isomorphism	
5.2. Complex graph operations	
5.2.1. Union	
5.2.2. Sum of two graphs	

5.2.3. Intersection	
5.2.4. Graph join	
5.2.5. Difference of graphs	
5.2.6. Graph compliment	
5.2.7. Power graph	
<b>6. Walks, paths and circuits</b>	<b>24-25</b>
6.1. Walk	
6.2. Closed walk	
6.3. Trail	
6.4. Circuit	
6.5. Path	
6.6. Cycle	
6.7. Connected graph	
6.8. Component	
<b>7. Representation of graphs as Matrix</b>	<b>25-26</b>
7.1. Adjacency matrix	
7.2. Incidence matrix	
<b>8. Graph in Circuits</b>	<b>26-30</b>
8.1. Tree	
8.2. Matrix	
8.2.1. Incident matrix	
8.2.2. Loop matrix or Tie set matrix	
8.2.3. Cut set matrix	
<b>9. Applications in Computer science</b>	<b>31-33</b>
9.1. Network	
9.2. Webpage	
9.3. Workflow	
9.4. Neural networks	
9.5. Google maps	
<b>10. Fingerprint Recognition using Graph Representation</b>	<b>33-37</b>
10.1. Finger print types	
10.2. Minute, core and delta	
10.3. Different classifications	
10.4. Old method, New method and Process	
10.5. Constriction of related weight graph	
10.5.1. Some information can be used in constructing the graph related to a finger print like	
10.5.2. Weight-age to Nodes and Edges	

<b>11. Graph theory models in security</b>	<b>38-40</b>
11.1. The four-color graph theorem	
11.1.1. Applying of the four color theorem in wireless a cell tower placement plan.	
11.1.2. Node coloring theorem	
<b>12. Network coding</b>	<b>40-42</b>
<b>13. Chemical graph theory</b>	<b>42-44</b>
13.1. Definitions	
13.2. Molecular energy	
13.3. Graph nullity and zero-Energy states	
<b>14. Recent applications of graph theory in molecular biology</b>	<b>45-48</b>
14.1. New applications in molecular biology	
14.2. New field has recently emerged called <i>bioinformatics</i> – application of IT and CS to molecular biology.	
<b>15. Knight's Tour</b>	<b>47-50</b>
<b>16. Common Problem</b>	<b>51-54</b>
16.1. House of Santa Claus	
16.2. Three utilities problem	
16.3. Seven Bridges of Königsberg	
16.4. Shortest path problem	
16.5. Travelling salesman problem(TSP)	
<b>17. Bibliography</b>	<b>55</b>

# **1 Graph theory**

In mathematics, **graph theory** is the study of *graphs*, which are mathematical structures, used to model pair wise relations between objects. A graph in this context is made up of *vertices*, *nodes*, or *points* which are connected by *edges*, *arcs*, or *lines*. A graph may be *undirected*, meaning that there is no distinction between the two vertices associated with each edge, or its edges may be *directed* from one vertex to another

## **1.1 History**



### The Königsberg Bridge problem

In the present century, there have already been a great many rediscoveries of graph theory which can only mention most briefly.

Euler (1707-1782) becomes the father of graph theory as well as Topology. The paper written by Leonhard Euler on the Seven Bridges of Königsberg and published in 1736 is regarded as the first paper in the history of graph theory. This paper, as well as the one written by Vandermonde on the knight problem, carried on with the analysis situs initiated by Leibniz. Euler's formula relating the number of edges, vertices, and faces of a convex polyhedron was studied and generalized by Cauchy and L'Huilier, and represents the beginning of the branch of mathematics known as topology.



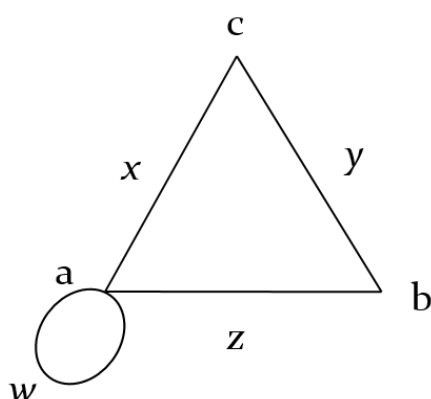
More than one century after Euler's paper on the bridges of Königsberg and while Johann Benedict Listing was introducing the concept of topology, Cayley was led by an interest in particular analytical forms arising from differential calculus to study a particular class of graphs, the trees. This study had many implications for theoretical chemistry.

## 1.2 Definitions

There is large number of definitions in graph theory. The following are some of the more basic ways of defining graphs and related mathematical structure

### Graph

a **graph** is an ordered pair  $G = (V, E)$  consisting of a non-empty set  $V = \{v_1, v_2, v_3, \dots\}$  of *vertices* or *nodes* or *points* together with a set  $E = \{e_1, e_2, e_3, \dots\}$  of *edges* or *arcs* or *lines*, which are 2-element subsets of  $V$  (i.e. an edge is associated with two vertices, and that association takes the form of the unordered pair comprising those two vertices) in other words A graph  $G$  is a collection of vertices “ $V$ ” and edges “ $E$ ”. So we say  $G = (V, E)$  is graph



Here vertices are a,b,c and  $a,b,c \in V$   
And edges are  $w,x,y,z$  and  $w,x,y,z \in E$

The set  $V(G)$  is called the **vertex set** of  $G$  and  $E(G)$  is called the **edge set**. Usually the **Graph** is denoted by  $G=(V,E)$  for  $u,v \in V$  set and  $\{u,v\}$  an edge of  $G$ . Since  $\{u,v\}$  is 2-elements set, we may write  $\{v,u\}$ .

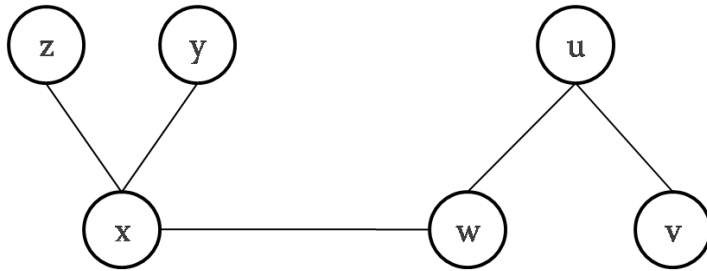
If  $e=uv$  is an edge of  $G$ , then we say that  $u$  and  $v$  are **adjacent** in  $G$  and  $e$  joins  $u$  and  $v$ .

**For example:**

A graph  $G$  is denoted by the sets

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

Now we have the following graph by considering these sets.



Every graph as a diagram associated with it.

If two distinct edges say  $e_1$  and  $e_2$  are **incident** with 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.

## **2 TERMINOLOGY**

**2.1 Loop:** an edge of the 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  $v_i = v_j$ .

**2.2 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 graph is called **multigraph**

**2.3 Directed & Un-directed graph:** If each edge of graph  $G$  has a direction then the graph is called **Directed Graph** (Example of directed graph fig.1). In **Directed Graph** each edge is represented by an arrow or direction curve from initial point  $u$  of  $e$  to the terminal point  $v$  (fig.2). If each edge of  $G$  has no-direction then the graph is called as An **Un--Directed Graph**.

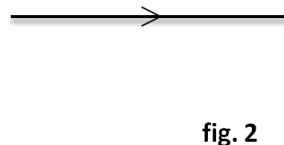
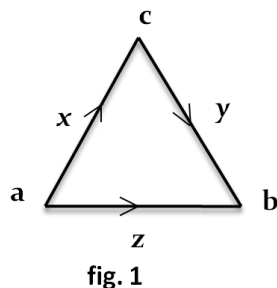
**2.4 Pseudo graph:** a graph with loops and multiple edges are allowed, is called a **pseudo graph**

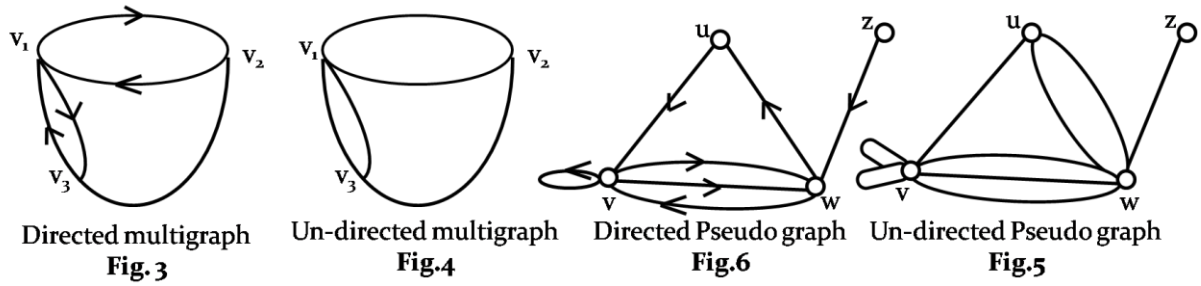
**2.5 Simple graph:** a graph which has neither loops nor multiple edges

**2.6 Finite and Infinite graphs:** a graph with finite number of vertices as well as a finite number of edges is called **finite graph**. Otherwise **infinite graph**.

**2.7 Degree of vertex:** The number of edges incident on a vertex  $v_j$  with self-loops is called the **degree of a vertex  $v_j$**  and denoted by  $\deg_G(v_j)$  or  $\deg v_j$  or  $d(v_j)$ .

**2.8 Isolated and Pendent vertices:** A vertex has **no incident edge** is called an **Isolated vertex**. In other words, isolated vertices are those with zero degree. A vertex of degree one is called **Pendent vertex** or **End vertex**.





## 1. Statement: (Handshaking theorem)

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

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

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

**2. Statement:** The number of vertices of odd degree in a graph is always even

**Problem 2.1.** Determine the number of edges in a graph with 6 vertices, 2 of degree 4 and 4 of degree 2. Draw two such graphs.

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

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

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

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

Hence the above equation,

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

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

Hence the number of edges in the graph with 6 vertices with given condition is 8. Two such graphs are shown below in figure 7.

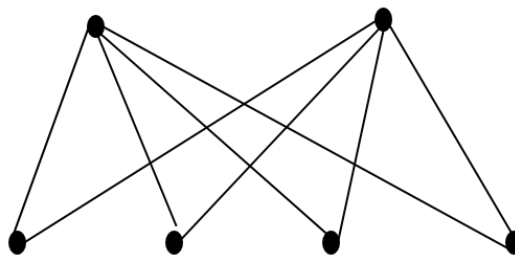
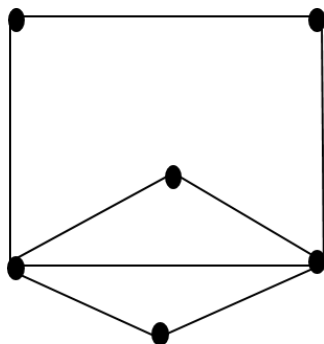
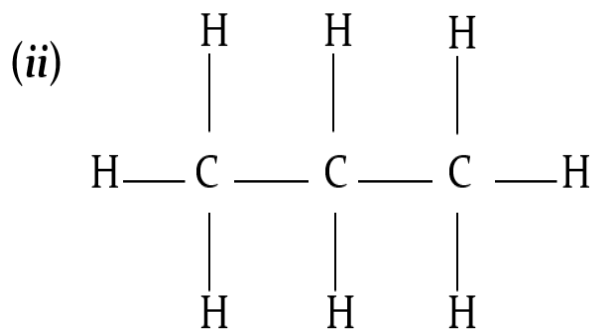
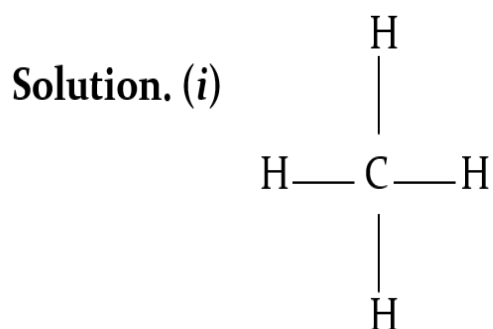


fig. 7

**Problem 2.2.** Draw the graphs of the chemical molecules of  
 (i) Methane ( $CH_4$ )                      (ii) Propane ( $C_3H_8$ )



### 3 Types of Graphs

**3.1 Null Graph:** A graph which contains only **isolated node**, is called a null graph

*i.e.*, the set of edges in the graph is empty.

Null graph is denoted on  $n$  vertices by  $N_n$

$N_4$  is shown in the finger 8.

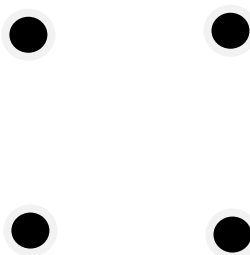


Fig.8

**3.2 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 complete graph is usually denoted by  $K_n$ . It should be noted that  $K_n$  has exactly  $\frac{n(n-1)}{2}$  edges.

The graph  $K_n$  for  $n= 1, 2, 3,4,5,6$  are shown in Fig 9.

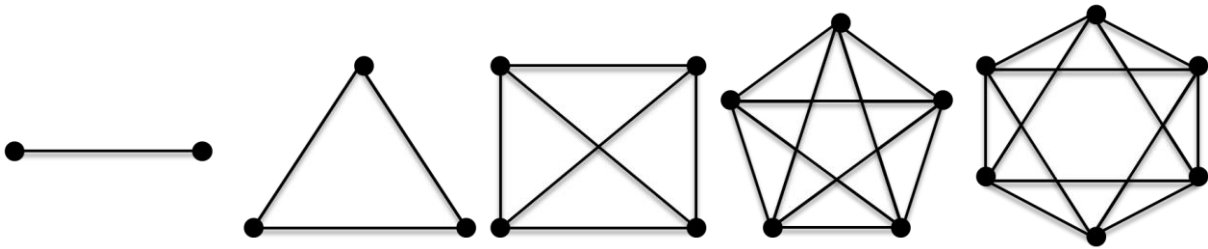


Fig.9

**3.3 Regular Graph:** A graph, in which all the 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  $r$ .

**3.4 Cycles:** The cycle  $C_n$ , consist of  $n$  vertices  $v_1, v_2, \dots, v_n$  and edges  $[v_1, v_2], [v_2, v_3], [v_3, v_4], \dots, [v_{n-1}, v_n]$ .

The cycles  $C_3, C_4, C_5$  and  $C_6$  are shown in Fig 10.

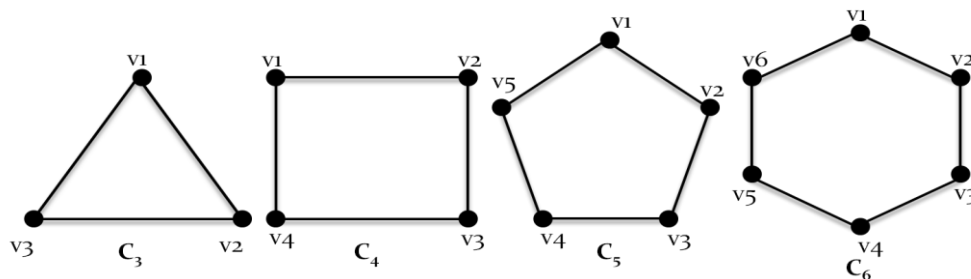


Fig.10

**3.5 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 the Fig 11.

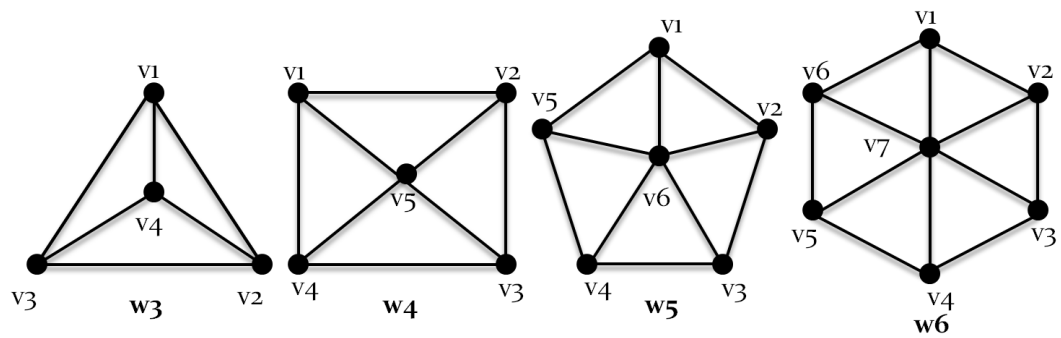
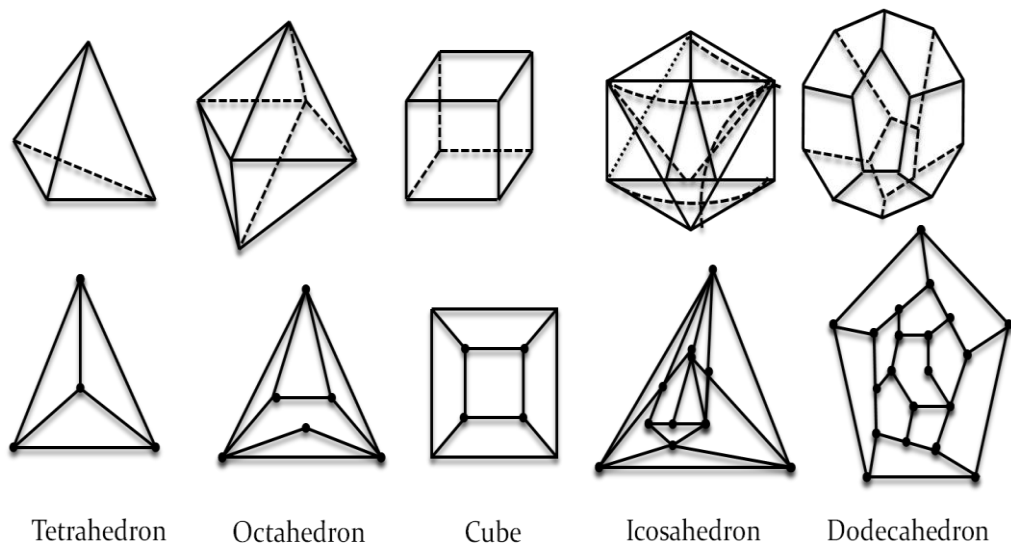


Fig.11

**3.6 Platonic Graph:** the graph formed by edges and vertices of five regular(platonic) solids-The tetrahedron, octahedron, cube, dodecahedron and icosahedrons.



Tetrahedron

Octahedron

Cube

Icosahedron

Dodecahedron

Fig.12

**3.7 N-cube :** The N-cube denoted by  $Q_n$ , is the graph that has vertices representing the  $2^n$  bit strings of length  $n$ . The 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 the figure 13. Thus  $Q_n$  has  $2^n$  vertices and  $n \cdot 2^{n-1}$  edges, and is regular of degree  $n$ .

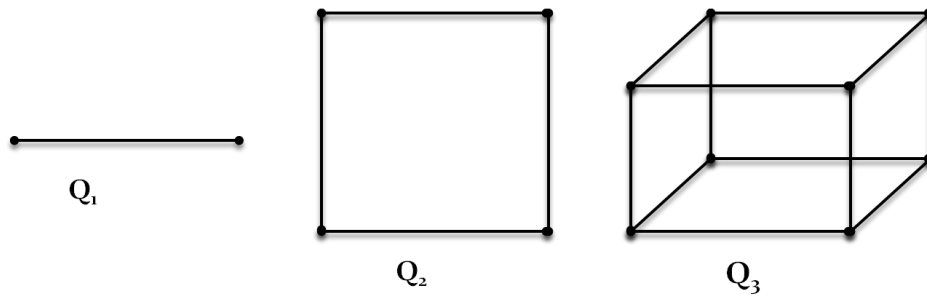


Fig.13

## 4 Tree and Forest

### 4.1 Tree

A *tree* is an undirected graph  $G$  that satisfies any of the following equivalent conditions:

- $G$  is connected and has no cycles.
- $G$  is acyclic, and a simple cycle is formed if any edge is added to  $G$ .
- $G$  is connected, but is not connected if any single edge is removed from  $G$ .
- $G$  is connected and the 3-vertex complete graph  $K_3$  is not a minor of  $G$ .
- Any two vertices in  $G$  can be connected by a unique simple path.

If  $G$  has finitely many vertices, say  $n$  of them, then the above statements are also equivalent to any of the following conditions:

- $G$  is connected and has  $n - 1$  edges.
- $G$  has no simple cycles and has  $n - 1$  edges.

As elsewhere in graph theory, the order-zero graph (graph with no vertices) is generally excluded from consideration: while it is vacuously connected as a graph (any two vertices can be connected by a path), it is not 0-connected (or even  $(-1)$ -connected) in algebraic topology, unlike non-empty trees, and violates the "one more vertex than edges" relation.

An **internal vertex** (or **inner vertex** or **branch vertex**) is a vertex of degree at least 2. Similarly, an **external vertex** (or *outer vertex*, *terminal vertex* or *leaf*) is a vertex of degree 1.

An *irreducible tree* (or *series-reduced tree*) is a tree in which there is no vertex of degree 2.



## 4.2 Forest

A *forest* is an undirected graph, all of whose connected components are trees; in other words, the graph consists of a disjoint union of trees.

Equivalently, a forest is an undirected acyclic graph. As special cases, an empty graph, a single tree, and the discrete graph on a set of vertices (that is, the graph with these vertices that has no edges), are examples of forests. Since for every tree  $V - E = 1$ , we can easily count the number of trees that are within a forest by subtracting the difference between total vertices and total edges.  $TV - TE = \text{number of trees in a forest}$ .

## 4.3 Poly tree

A *polytree*<sup>[11]</sup> (or *oriented tree*<sup>[4A][4B]</sup> or *singly connected network*<sup>[4C]</sup>) is a directed acyclic graph (DAG) whose underlying undirected graph is a tree. In other words, if we replace its directed edges with undirected edges, we obtain an undirected graph that is both connected and acyclic.

A *directed tree* is a directed graph which would be a tree if the directions on the edges were ignored, i.e. a polytree. Some authors restrict the phrase to the case where the edges are all directed towards a particular vertex, or all directed away from a particular vertex (see arborescence)

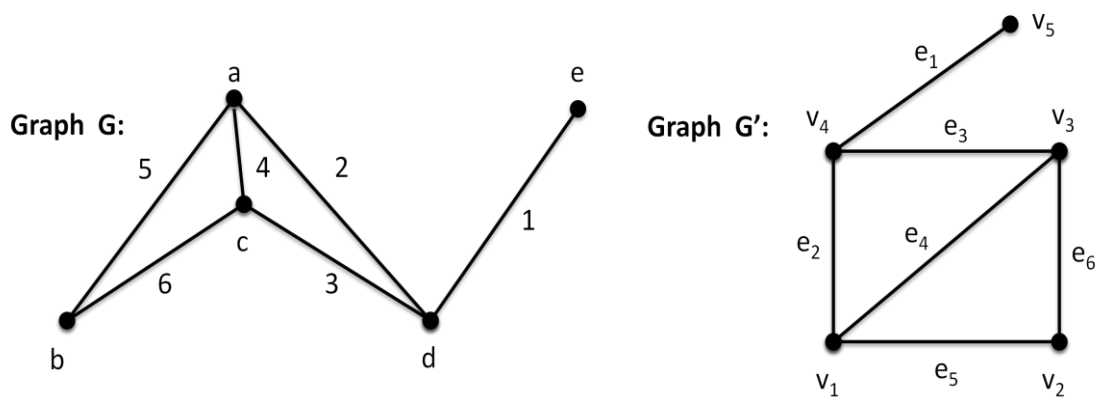
## 4.4 Types of Trees

- A path graph (or *linear graph*) consists of  $n$  vertices arranged in a line, so that vertices  $i$  and  $i+1$  are connected by an edge for  $i=1, \dots, n-1$ .
- A starlike tree consists of a central vertex called *root* and several path graphs attached to it. More formally, a tree is starlike if it has exactly one vertex of degree greater than 2.
- A star tree is a tree which consists of a single internal vertex (and  $n-1$  leaves). In other words, a star tree of order  $n$  is a tree of order  $n$  with as many leaves as possible.
- A caterpillar tree is a tree in which all vertices are within distance 1 of a central path subgraph.
- A lobster tree is a tree in which all vertices are within distance 2 of a central path subgraph.

## 5 GRAPH ISOMORPHISM and GRAPH OPERATIONS:

### 5.1 Graph isomorphism:

Two graphs  $G$  and  $G'$  are said to be **isomorphic** to each other if there is a one-to-one correspondence (bijection) between their vertices and between their edges such that the incidence relationship is preserved.



Correspondence of vertices

$$f(a) = v_1$$

$$f(b) = v_2$$

$$f(c) = v_3$$

$$f(d) = v_4$$

$$f(e) = v_5$$

Correspondence of edges

$$f(1) = e_1$$

$$f(2) = e_2$$

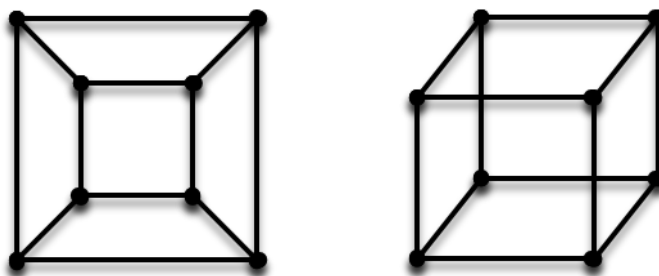
$$f(3) = e_3$$

$$f(4) = e_4$$

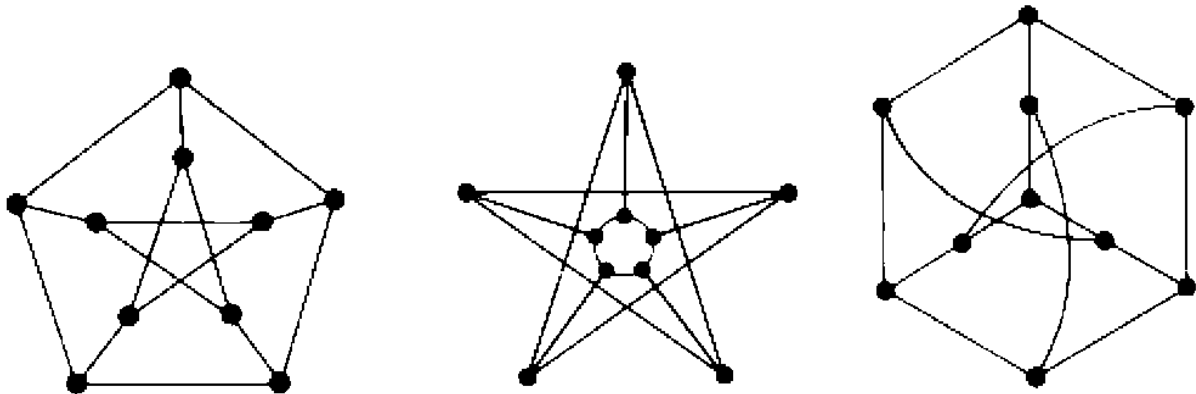
$$f(5) = e_5$$

Adjacency also preserved. Therefore  $G$  and  $G'$  are said to be isomorphic.

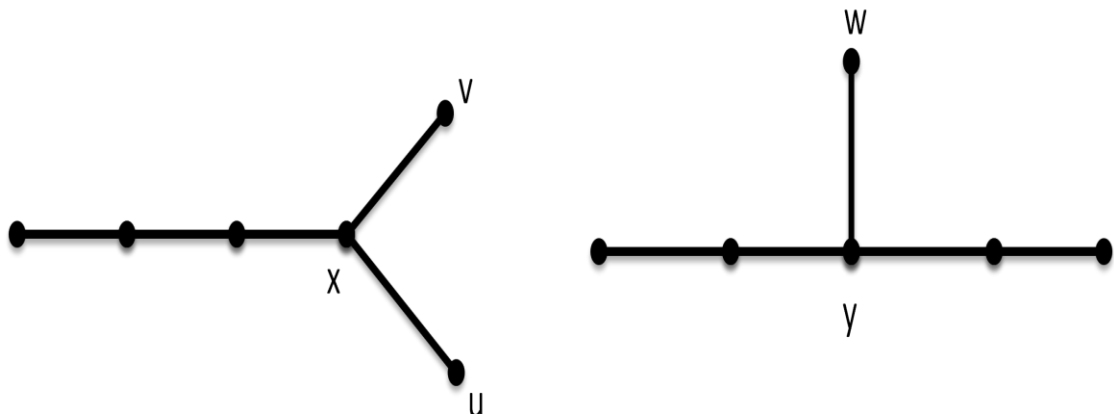
The following graphs are isomorphic to each other. i.e two different ways of drawing the same graph



The following three graphs are isomorphic.



The following two graphs are not isomorphic, because  $x$  is adjacent to two pendent vertex is not preserved.

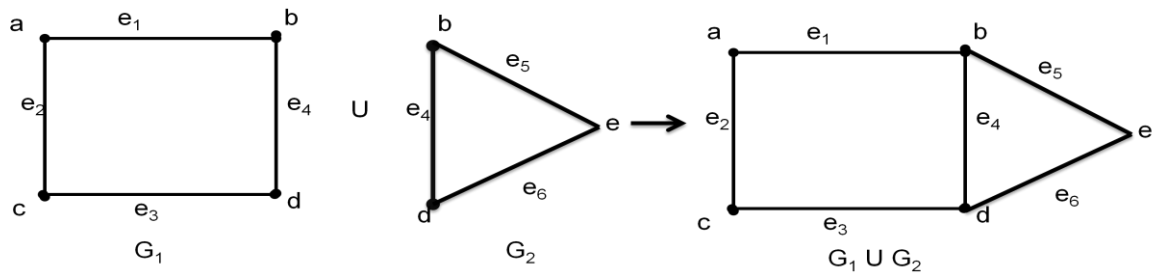


## 5.2 Complex Graph Operations:

**5.2.1 Union :** given two graphs  $G_1$  and  $G_2$ , their union will be a graph

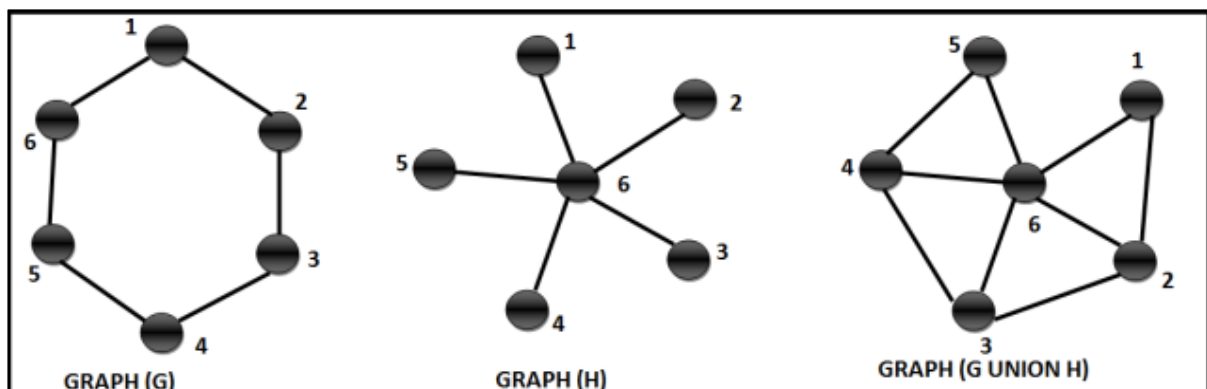
such that  $V(G_1 \cup G_2) = V(G_1) \cup V(G_2)$

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



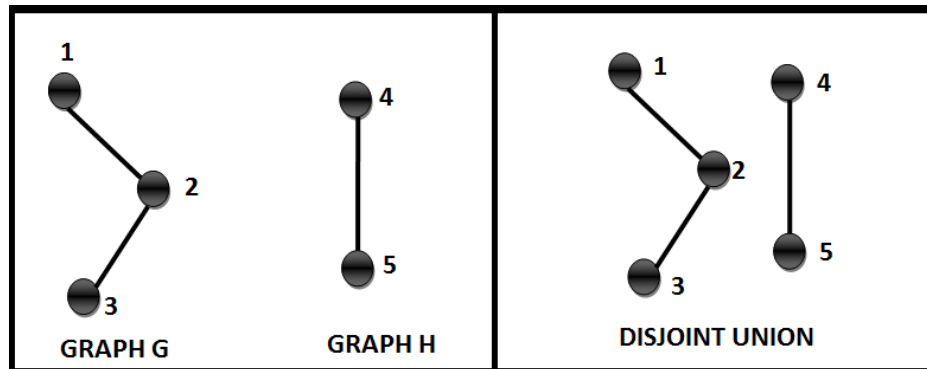
The union of two graphs  $G(V_G, E_G)$  and  $H(V_H, E_H)$  is the union of their vertex sets and their edge families. Which means  $G \cup H = (V_G \cup V_H, E_G \cup E_H)$ .

The below image shows a union of graph  $G$  and graph  $H$ . Make sure that all the vertices and edges from both the graphs are present in the union. If possible draw it once yourself.



**5.2.2 Sum of two graphs:** In graph theory, a branch of mathematics, the **disjoint union of graphs**[6B] is an operation that combines two or more graphs to form a larger graph. It is analogous to the disjoint union of sets, and is constructed by making the vertex set of the result be the disjoint union of the vertex sets of the given graphs, and by making the edge set of the result be the disjoint union of the edge sets of the given graphs. Any disjoint union of two or more nonempty graphs is necessarily disconnected. the

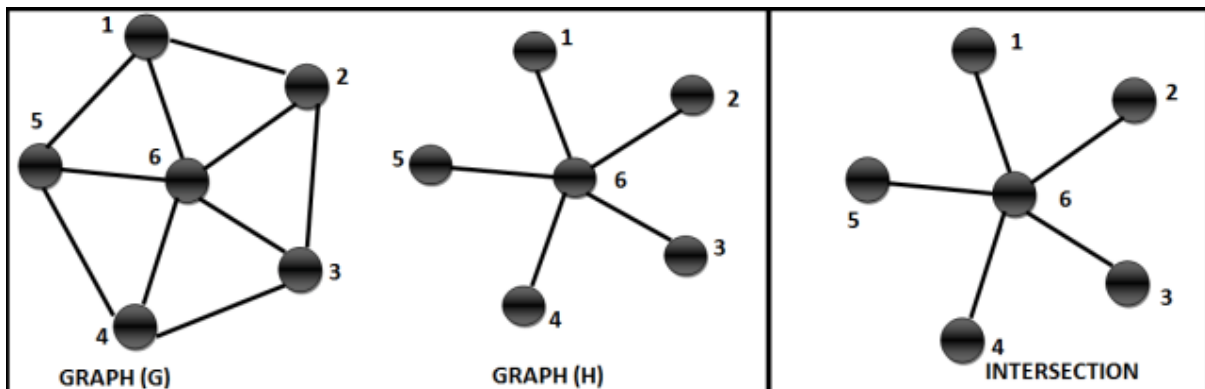
union where the  $V_G$  and  $V_H$  are disjoint. In this case the Union is referred to as disjoint Union and it is denoted by  $G + H$ .



**5.2.3 Intersection:** The intersection of two graphs  $G(V_G, E_G)$  and  $H(V_H, E_H)$  is the union of their vertex sets and the intersection of their edge families. Which means  $G \cap H = (V_G \cup V_H, E_G \cap E_H)$ .

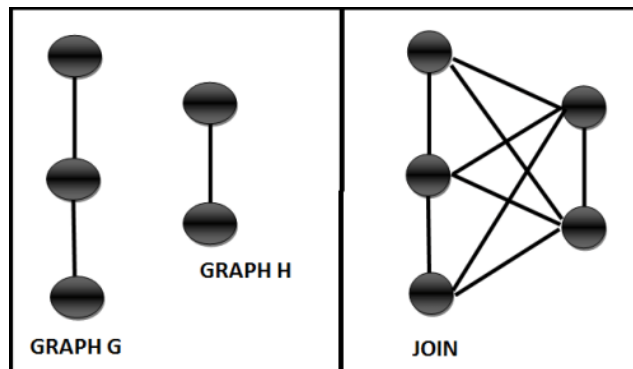
Below image demonstrate the Intersection of the two Graphs.

Notice that the vertex set is a union of both the vertex sets but the edge family consists only the edges which exist in both the graphs G and H.



**5.2.4 Graph join:** Given two graphs  $G(V_G, E_G)$  and  $H(V_H, E_H)$ , the join Graph will contain the union of the two graphs and all edges which can connect the vertices of G to vertices of H.

Below is an image to demonstrate that.



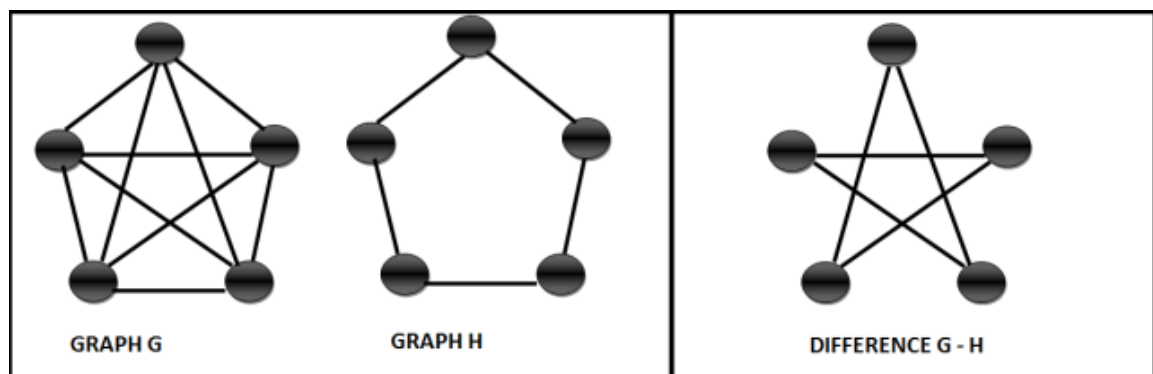
Note that the join graph contains new edges (which join the vertices of G to vertices of H) apart from the disjoint union.

**5.2.5 Difference of Graphs:** Given two graphs  $G(V_G, E_G)$  and  $H(V_H, E_H)$ , the difference Graph  $G - H$  will contain the union of the vertices of two graphs G and H and the edges in the difference is the difference of edges which is  $E_G - E_H$

Few points

- The Graph Difference of any graph and itself is an empty graph. Which means  $G - G = \text{Empty Graph}$ .
- The vertices of the Graph Difference ( $G - H$ ) are the union of the vertices of the graphs G and H
- The edges of the Graph Difference ( $G - H$ ) are the complement of the edges of the graphs G and H.
- The Graph Difference ( $G - H$ ) of two graphs has the same vertices as Graph Union ( $G \cup H$ )
- The Graph Difference ( $G - H$ ) of two graphs has the same vertices as Graph Intersection  $G \cap H$

Below is an image to demonstrate the Graph Difference (union of the two vertices)



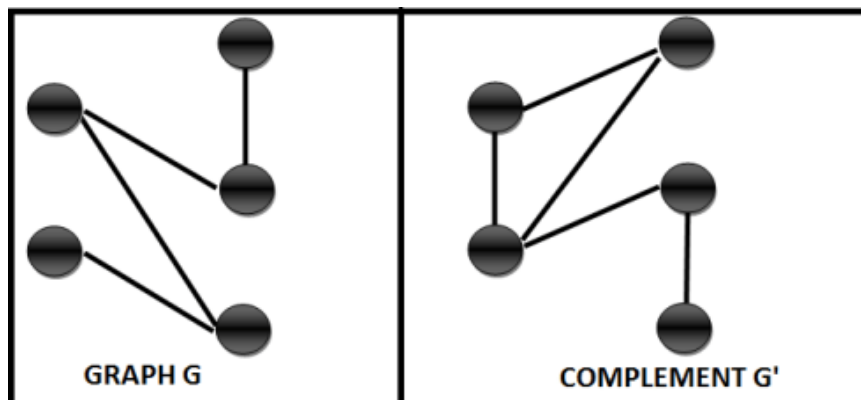
Note that the vertex set is the union of the two vertex sets where as the edges in the difference is the family of edges which are present in  $G$  but not in  $H$ .

**5.2.6 Graph Compliment:** The complement of graph  $G(V, E)$  is a graph that has the same vertices as  $G$  but the edges defined by two vertices in the complement is adjacent only if they are not adjacent in  $G(V, E)$ .

Few points

- The Graph complement  $G'$  is also called edge complement.
- Vertex set is same for both  $G$  and  $G'$ .
- Edges in  $G'$  are not present in  $G$ .
- The union of  $G$  and  $G'$  is a complete graph.
- The complement of a complete Graph is a Null Graph.

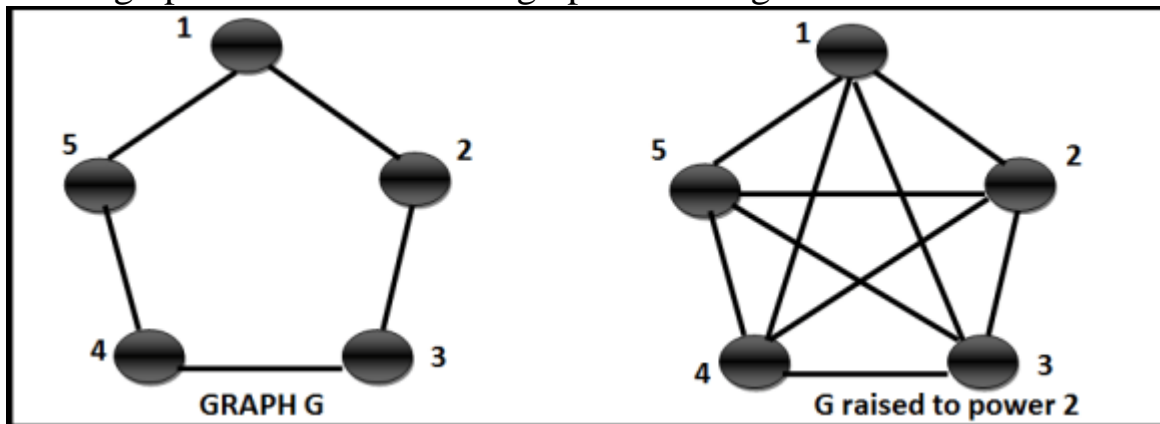
Below is a demonstration of Graph Complement. Note that the edges in  $G$  are not present in  $G'$



**5.2.7 Power Graph:** The Power Graph of Graph  $G(V, E)$  is always calculated with respect to a constant  $k$ . Which means we will always say that the  $k^{\text{th}}$  power of  $G$  and not just power of  $G$ . The  $k^{\text{th}}$  power of  $G$  is a graph with the same set of vertices as  $G$  and an edge between two vertices in the Power Graph exists only if there is a path of length at most  $k$  between them.

Consider the below graph  $G(V, E)$ , and let's say that we need to derive its square, which means the value of  $k$  is 2.

So the graph will result into the graph on the right.

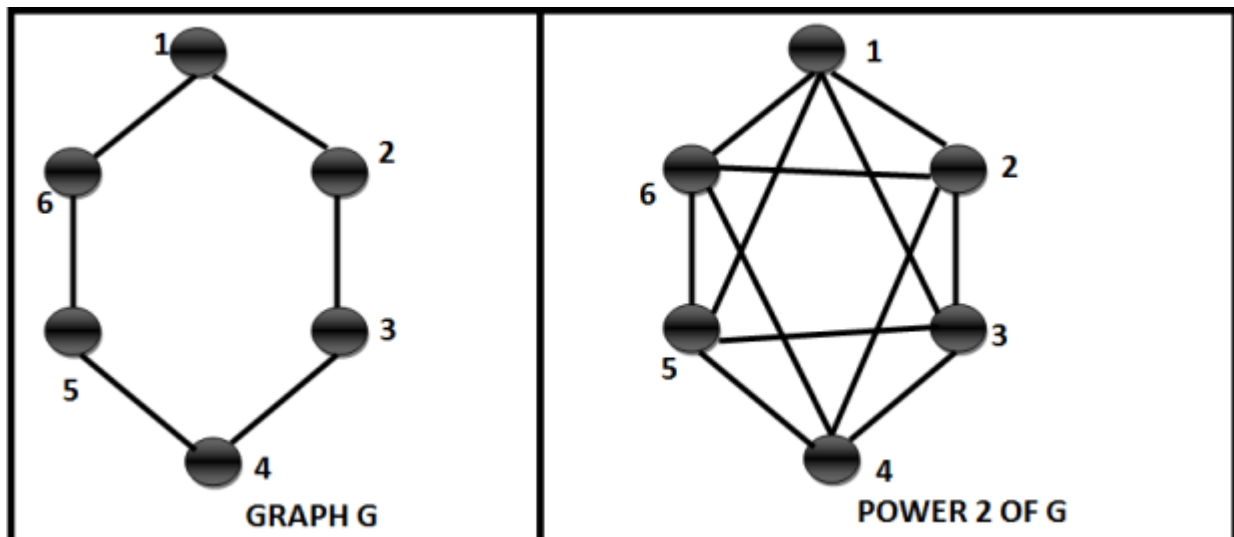


Note that the vertices 1 and 3 are connected because there exists a path of length two or less in the original graph..

Similarly there exists paths of length two or less between  $\{2, 4\}$ ,  $\{3, 5\}$ ,  $\{4, 1\}$ ,  $\{5, 2\}$ , hence these vertices are connected in the power graph.

This concludes that all the graph are the first power of themselves. Which means if value of  $k$  is 1 then we get the same graph? And if the adjacency matrix of  $G$  is  $A$  then each cell  $(i,j)$  of  $A^1$  will give you the number of paths of length 1 between  $i$  and  $j$ .

Lets take another example. Again let the value of  $k$  is 2. Consider the graph below.



The edges  $\{1,3\}$ ,  $\{2,3\}$ ,  $\{3,4\}$ ,  $\{4,6\}$ ,  $\{5,1\}$  and  $\{6,2\}$  are connected because they have a path length of 2 or less between them in the original graph.

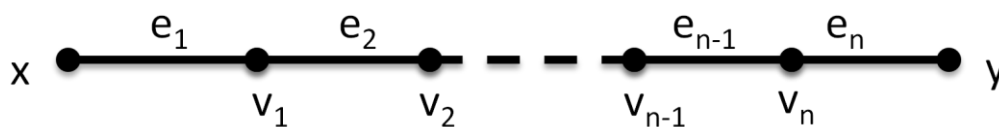
The edges  $\{1,4\}$ ,  $\{2,5\}$  and  $\{3,6\}$  are not connected because the length of the path between them is three which is more than two.



## 6 WALKS, PATHS, CIRCUITS

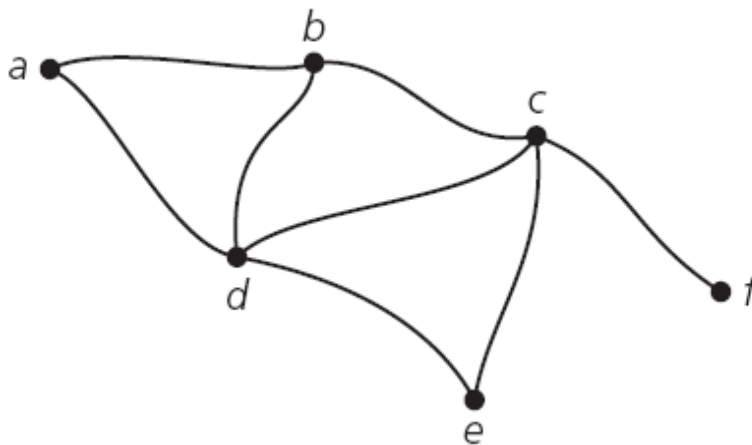
**6.1 Walk:** A **walk** is defined as a finite alternating sequence of vertices and edges, beginning and ending with vertices. No edge appears more than once. It is also called as an edge train or a chain.

Walk “x to y” is  $W = \{xe_1v_1e_2v_2\dots v_{n-1}e_nv_ny\}$  a sequence of vertices and edges.



**6.2 Closed Walk:** closed walk occurs when  $x=y$ (in above fig) i.e., a walk from vertex v to the same v.

**6.3 Trail :** a trail is a walk with no repeated edges



A walk from a to f = abcf , adcf ,adecf and abdcf are trails.

A walk from a to f = adbdcf is not a trail.

**6.4 Circuit:** A closed walk in which no edge appears more than once is called a **circuit**. That is, a circuit is a closed trail.

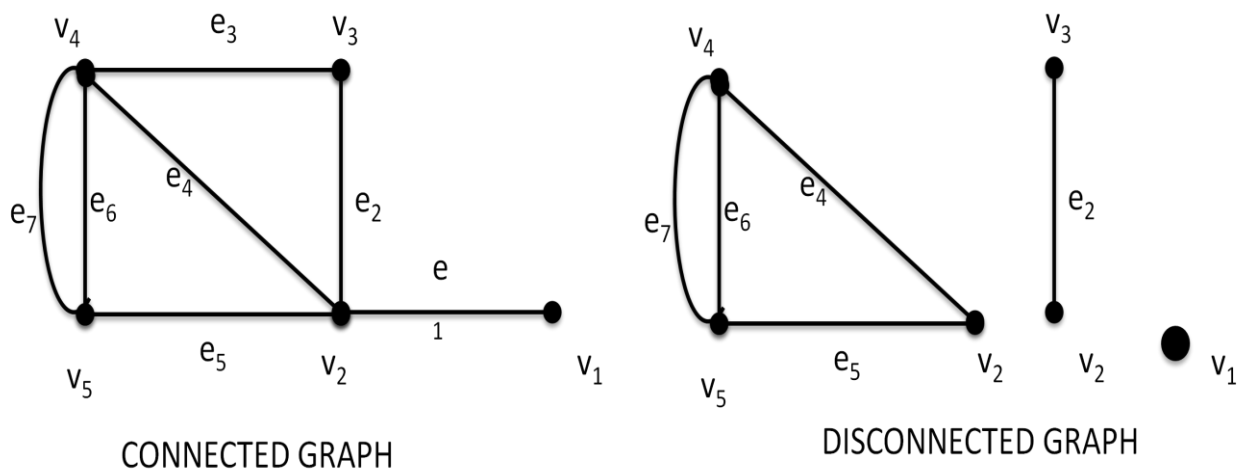
A walk from a to a = adeba ,adcba ,abceda and adb are circuits.

**6.5 Path:** An open walk in which no vertex appears more than once is called **path**. The number of edges in the path is called **length of a path**.

**6.6 Cycle:** A closed path is called as cycle

**6.7 Connected graph:** A graph  $G$  is said to be **connected** if there is at least one path between every pair of vertices in  $G$ . Otherwise,  $G$  is disconnected.

**6.8 Component:** disconnected graph consists of two or more connected graphs. Each of these connected subgraph is called a component.



## 7 Representation of graphs as Matrix:

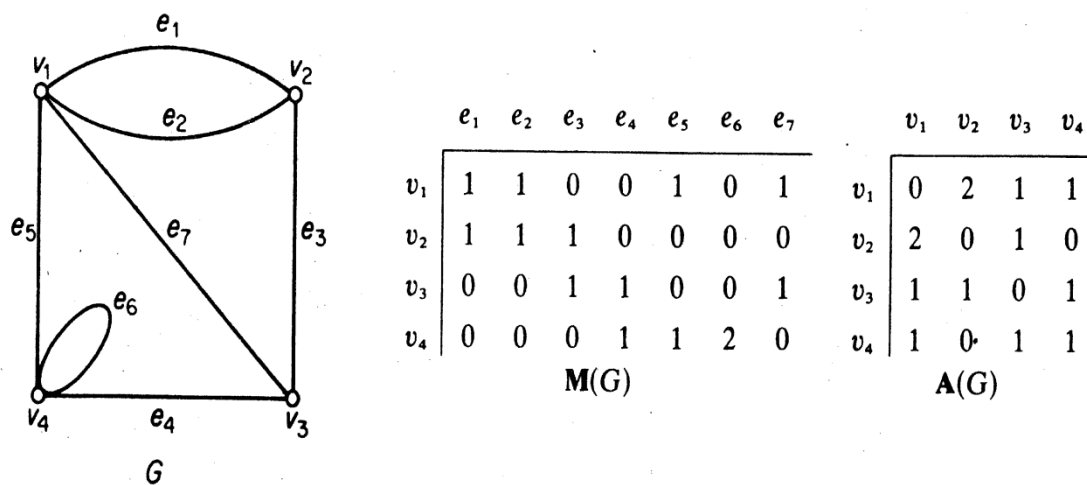
Although a diagrammatic of a graph is very convenient for visual study but this is only possible when the number of edges and vertices are reasonably small.

The matrix are commonly used to represent graphs for computer processing. The advantages of representing the graphs in matrix form lies on the fact that many result of matrix algebra can be readily applied to study the structures of graphs from an algebraic point of view.

Two types representation are given below:

**7.1 Adjacency matrix:** To any graph  $G$  there corresponds a  $v \times v$  matrix  $A(G) = [a_{ij}]$ , in which  $a_{ij}$  is the number of edges joining  $v_i$  and  $v_j$ .

**7.2 Incidence matrix:** To any graph  $G$  there corresponds a  $v \times e$  matrix called the incident matrix of  $G$ . The incidence matrix of  $G$  is the matrix  $M(G) = [m_{ij}]$ , where  $m_{ij}$  is the number of times (0, 1 or 2) that  $v_i$  and  $e_j$  are incident. this is a different way of specifying the graph.

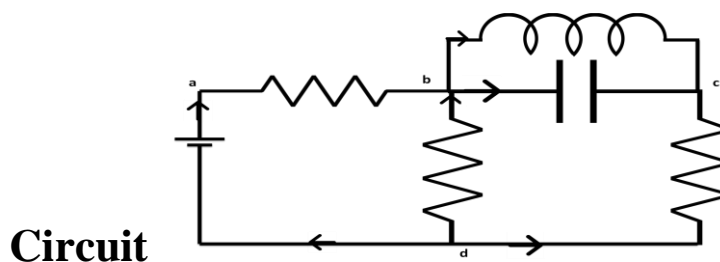


## 8 Graph in Circuits:

**Graph:** Diagram consisting Node and Branch is called graph.

**Node:** Inter connection of components.

**Branch:** Line connecting two nodes.

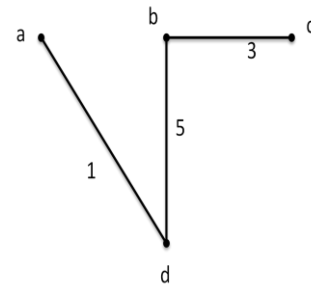
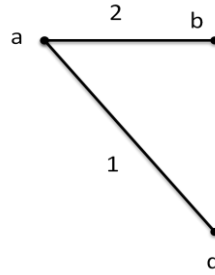
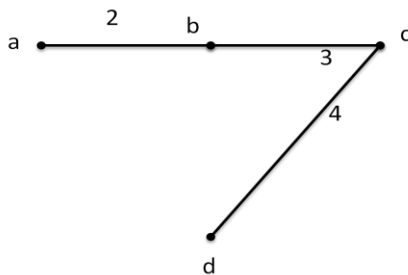
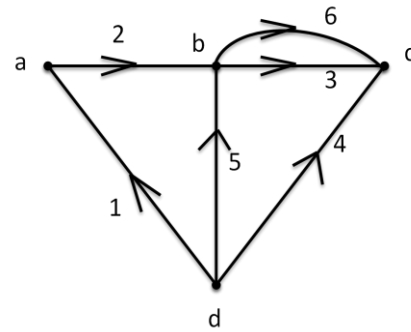


**Circuit**

Here a, b, c and d are called Nodes and ab, bc, cd, db, and ad are Branches.

When we convert this circuit in to graph we get

4-nodes (a, b, c, d)  
6-branches.



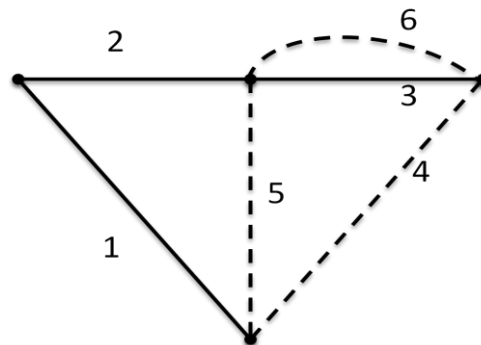
## 8.1 Tree : collection of nodes without any loop

There are 2 things in trees

- 1: **Twig:** Branches of a tree are called as twigs
- 2: **Link:** Remaining branches of graph after the tree taken are called as links

Twig: - 1, 2 and 3

Links: -4, 5 and 6



## 8.2 Matrix :-

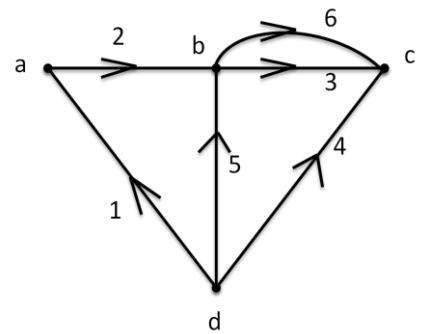
In graphs we have 3 kinds of matrices.

Here we take the values as per the direction. If the arrow is coming towards node we take -1 and if the arrow going away from node we take +1. if node and branch has no connection we take zero as a element in matrix corresponding to them.

### 8.2.1 Incident matrix:-

The incident matrix for the given graph is

Branches Nodes						
	1	2	3	4	5	6
a	-1	+1	0	0	0	0
b	0	-1	+1	0	-1	+1
c	0	0	-1	-1	0	-1
d	+1	0	0	+1	+1	0



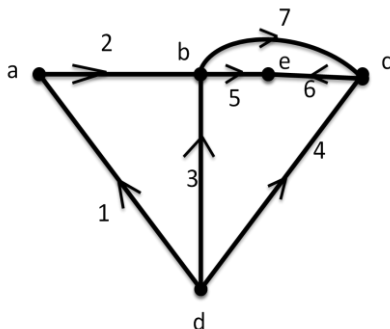
$[A_c]$  = Complete incident

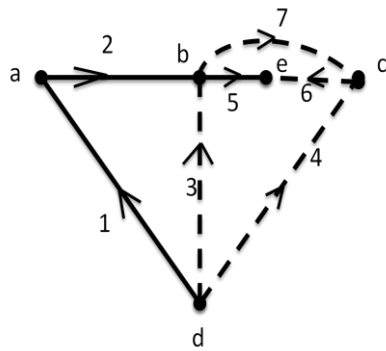
matrix.

(Since every Branch consist of one -1 and one +1.)

$[A]$  = Incident matrix. (if we delete the row d then we get the matrix is called incident)

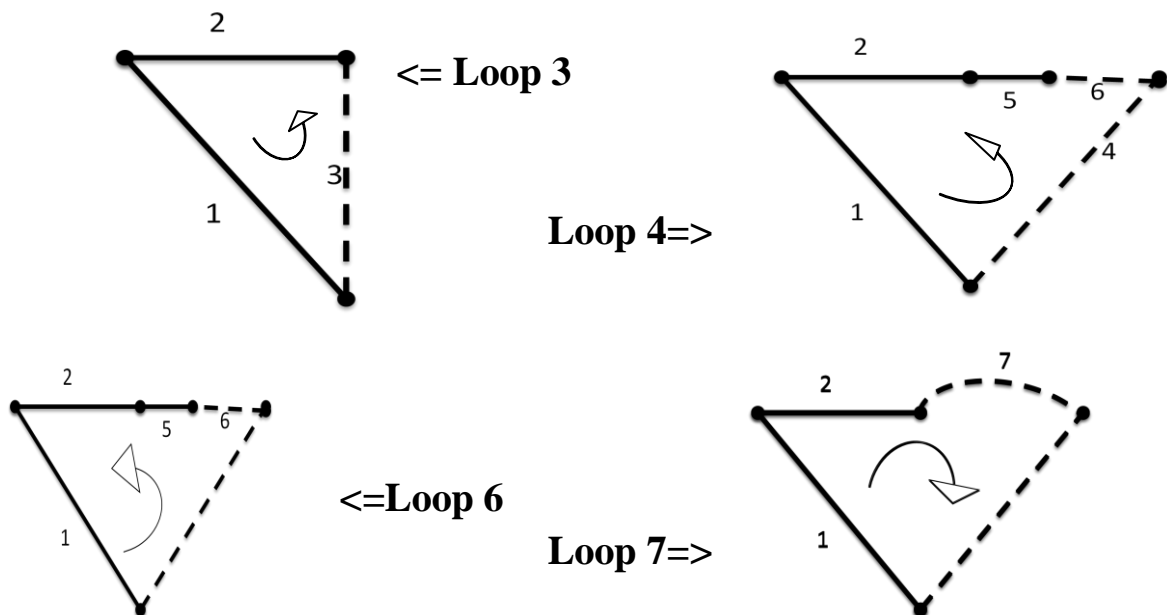
### 8.2.2 Loop matrix (or) Tie set matrix:-





Branches Loop	1	2	3	4	5	6	7
7	+1	+1	0	0	0	0	+1
6	-1	-1	0	0	-1	+1	0
4	-1	-1	0	+1	-1	0	0
3	-1	-1	+1	0	0	0	0

Here we will consider the loops existed in the graph. We will take those branches which leads to loops in graph. The following are the loops occurred in graph

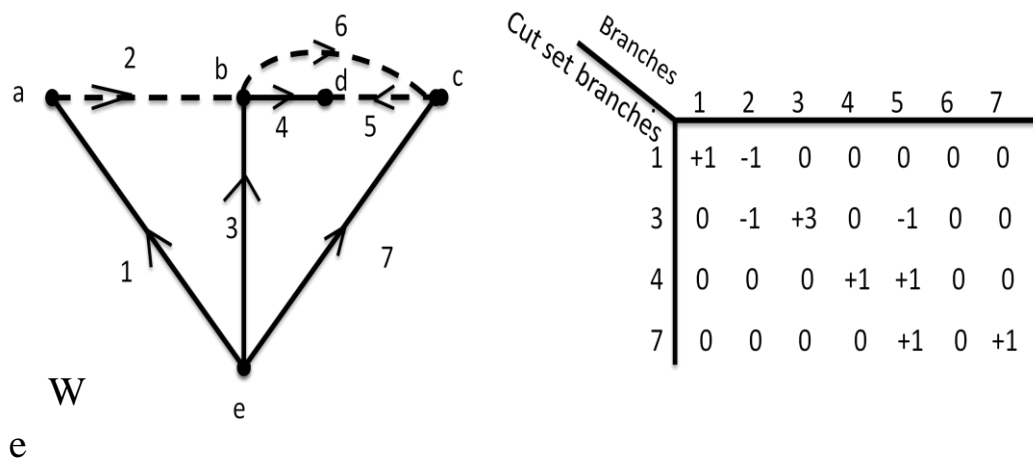


We can verify that the matrix is correct or not i.e.,  
The matrix between loops and itself gives identity matrix .

Branches					
Loops		3	4	5	6
	3	+1	0	0	0
	4	0	+1	0	0
	5	0	0	+1	0
	6	0	0	0	+1

### 8.2.3 Cut set matrix

By taking the cut set tree, we form a matrix between branches of cut set branches of original graph.

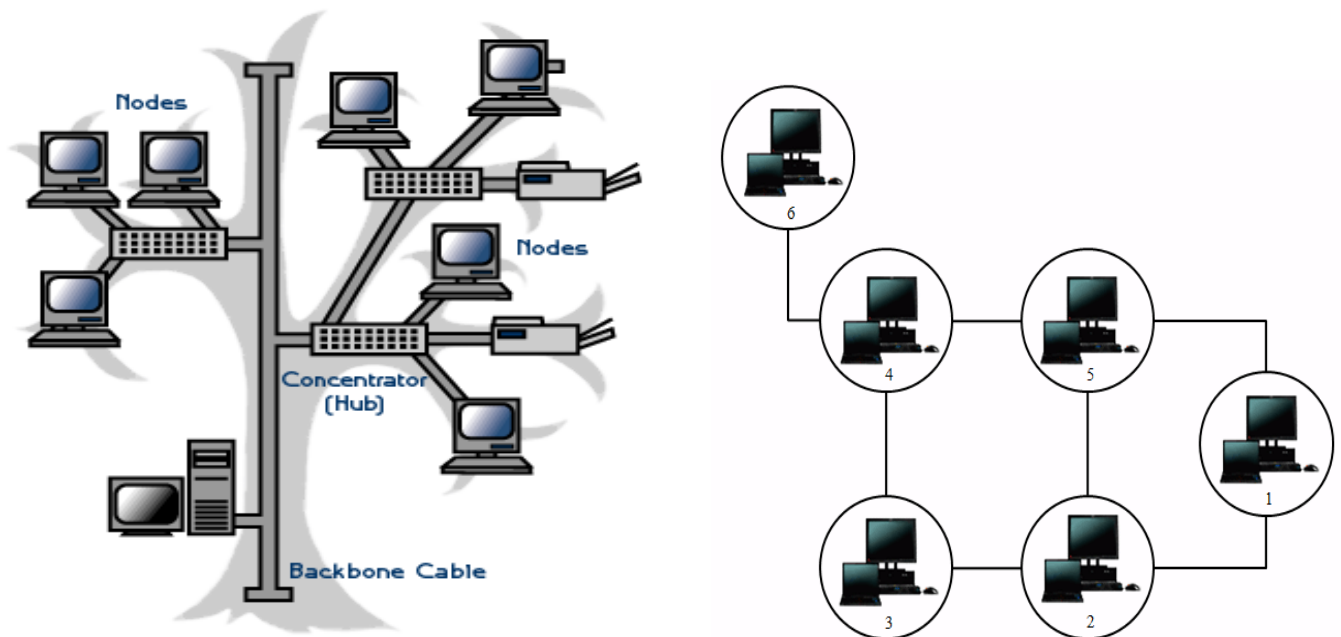


consider the cut set branches 1, 3, 4 and 7. Taking +1 if the branch of a column is in the direction of branch on row. If not we take -1 at which the node at the head of the branch is considered and takes zero if the branch is not connected to that node.

## 9 Applications in computer Science

Since computer science is not a concrete/centralized subject, we can introduce graph theory in many areas

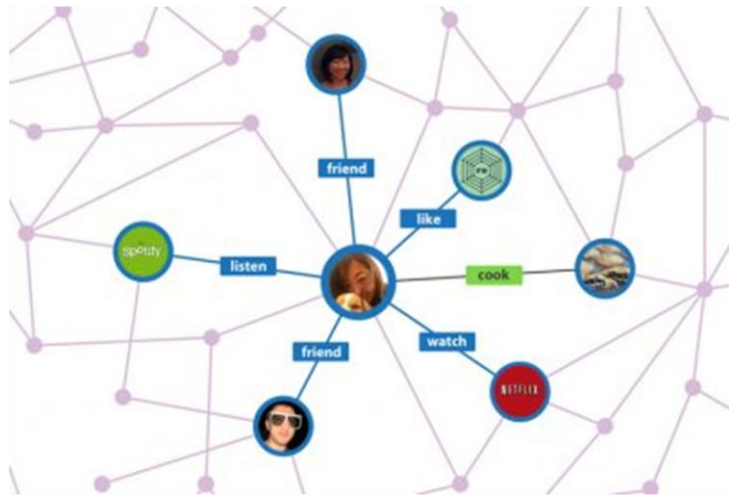
**9.1 Networks:** Graph theory can be used in computer networks, for security purpose or to schematize network topologies, for example.



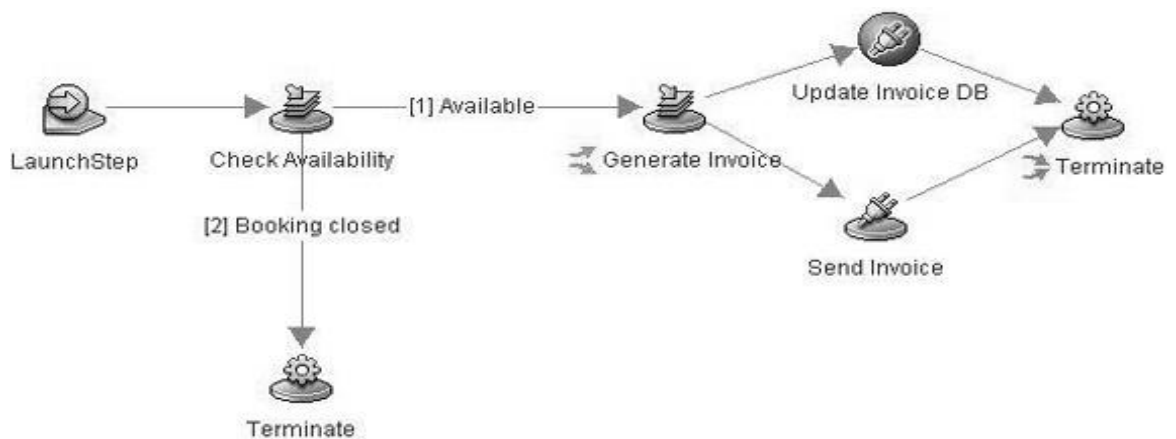
**9.2 Webpage:** can be represented by a directed graph. The vertices are the web pages available at the website and a directed edge from page  $A$  to page  $B$  exists if and only if  $A$  contains a link to  $B$



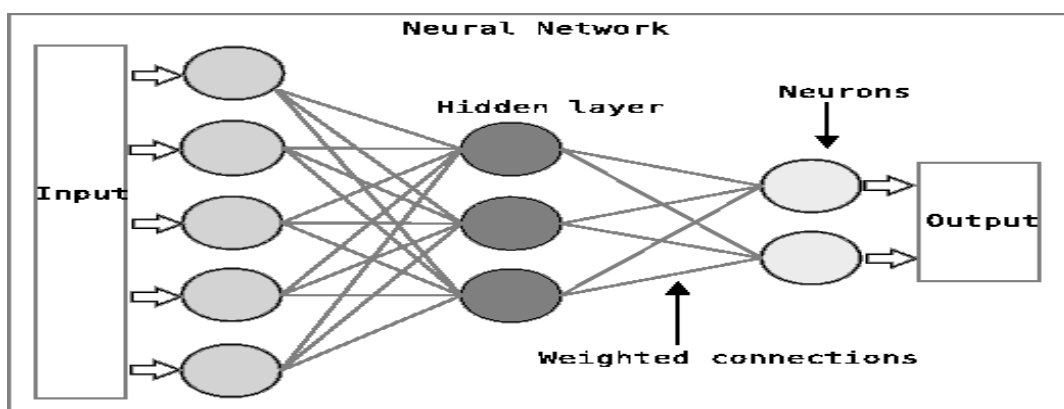
Facebook is based in graph theory



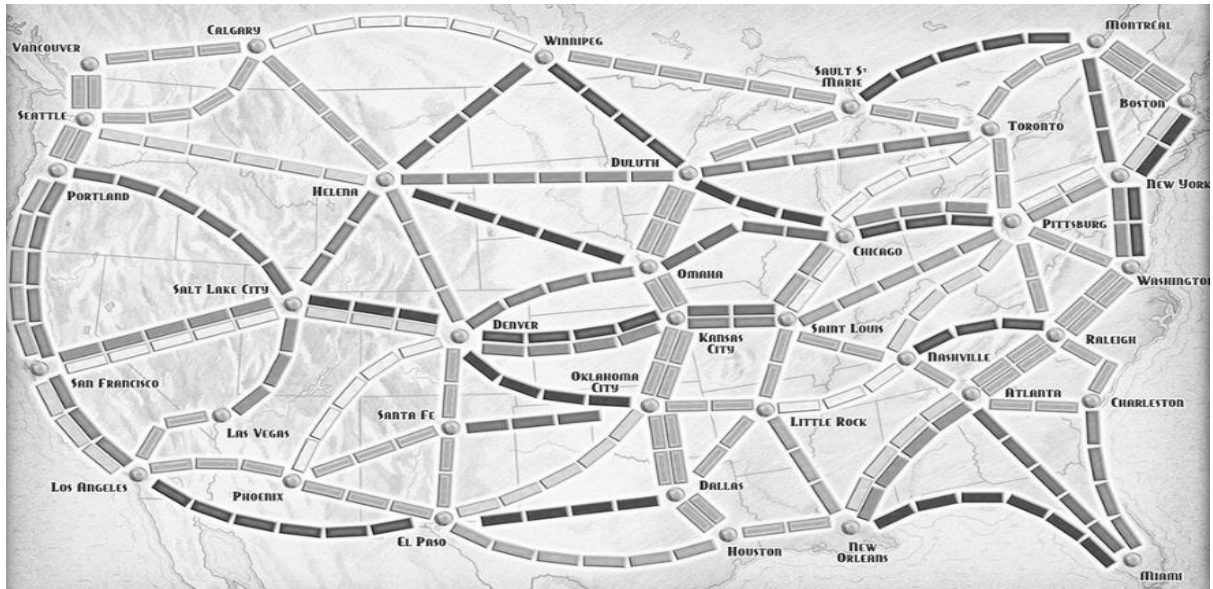
**9.3 Workflow:** It's sequence of processes through which a piece of work passes from initiation to completion. Can be also represented as directed graph.



**9.4 Neural Networks:** A series of algorithms that attempt to identify underlying relationships in a set of data by using a process that mimics the way the human brain operates.



## 9.5 Google Maps:



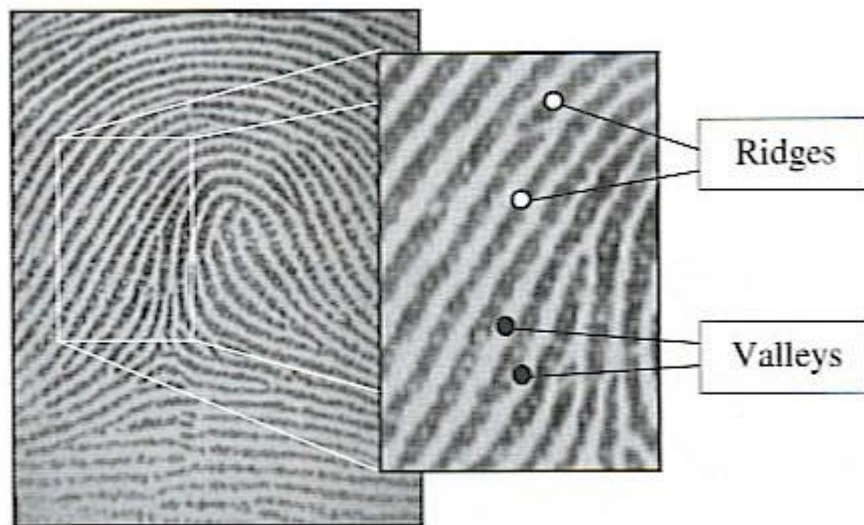
## 10 Fingerprint Recognition using Graph Representation

The three characteristics of FINGERPRINTS are:

1. There are no similar fingerprints in the world.
2. Fingerprints are unchangeable.
3. Fingerprints are one of the unique features for identification systems.










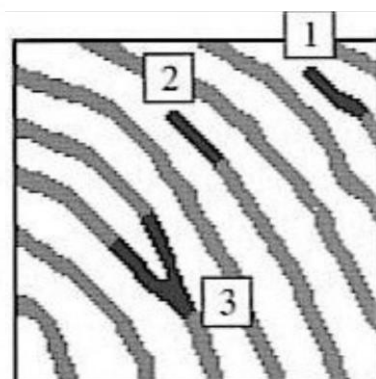
## 10.1 FINGERPRINT TYPES



The lines that flow in various patterns across fingerprints are called ***Ridges*** and the spaces between ridges are ***Valleys***.

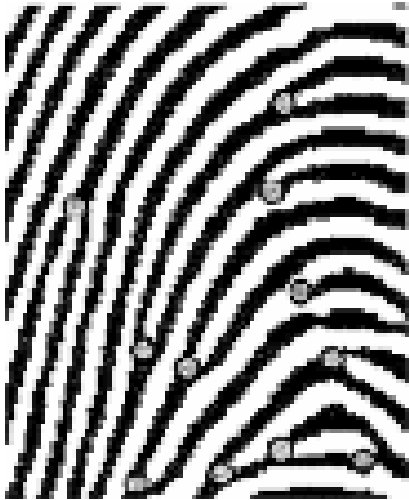
Types of patterns in Fingerprint.

	Termination
	Bifurcation
	Lake
	Independent ridge
	Point or island
	Spur
	Crossover



1 and 2 are terminations.  
3 is bifurcation.

## 10.2 MINUTE, CORE AND DELTA

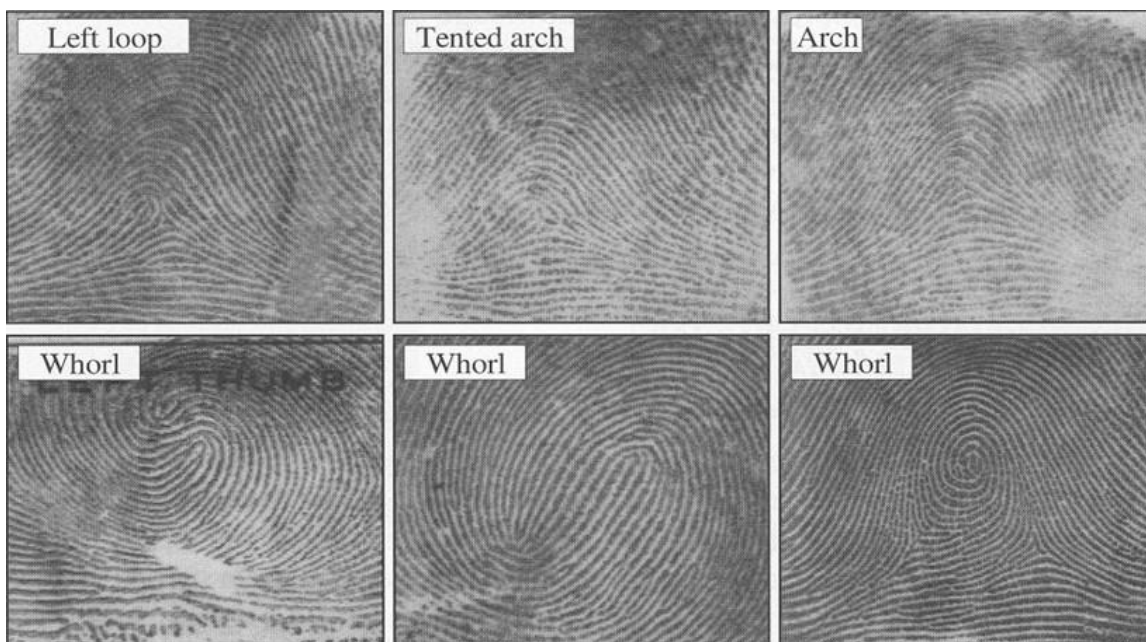


**Minute** – The places at which the ridge intersects or ends

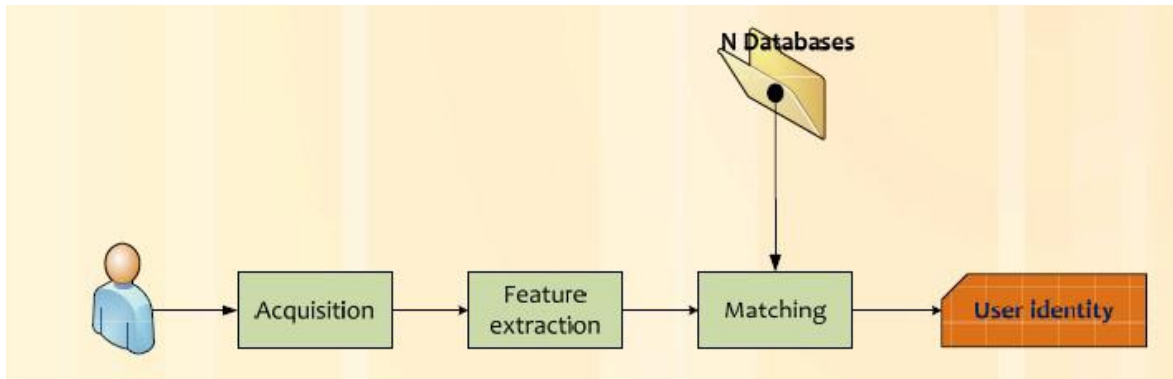
**Core** – The places where the ridges form a half circle.

**Delta** – The places where the ridges form a triangle.

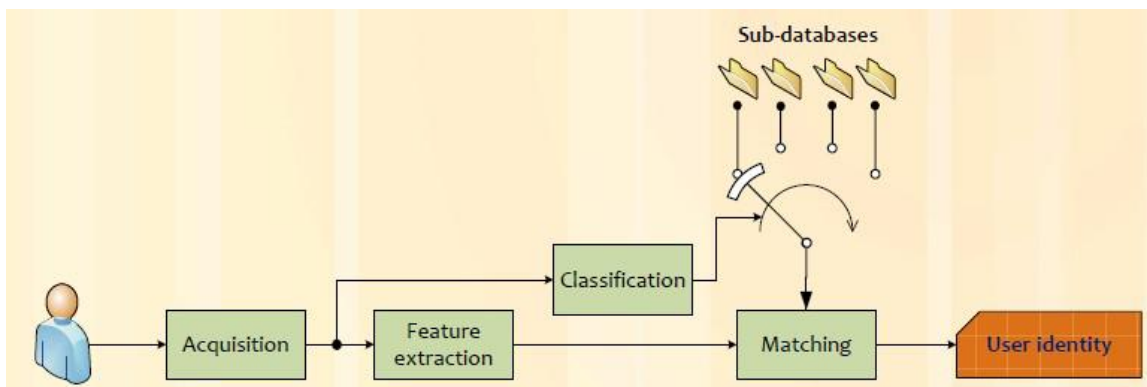
## 10.3 DIFFERENT CLASSIFICATION



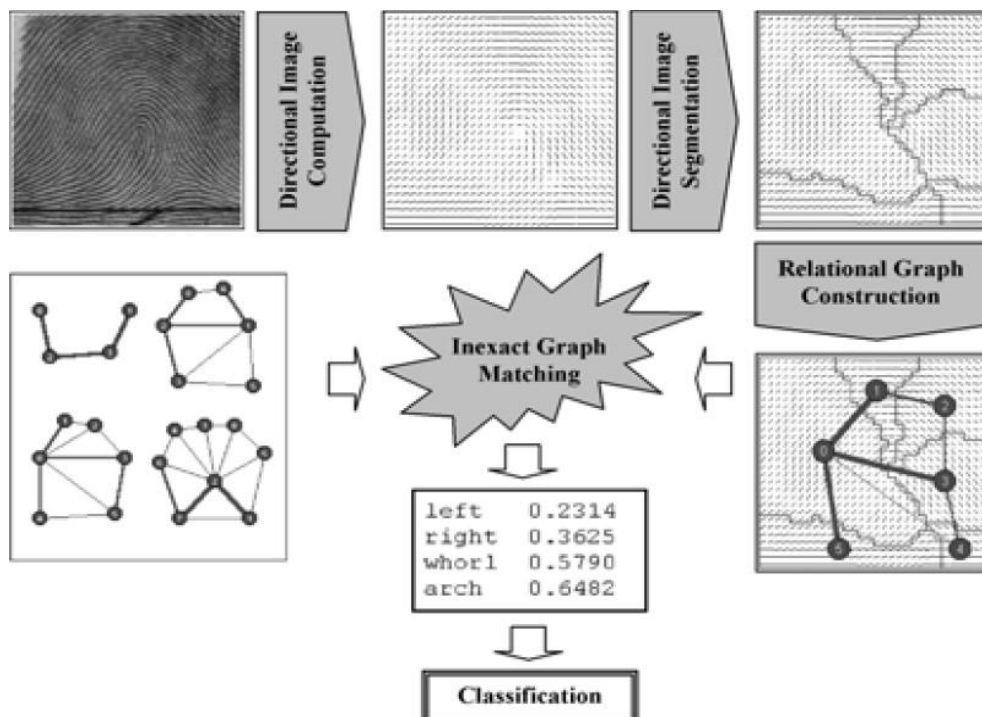
## 10.4 OLD METHOD:-



## NEW METHOD:-



## PROCESS FLOWS :-



## 10.5 Construction of related weight graph

Graph can be shown by G index including four parameters of  $G = (V, E, \mu, v)$ . Where  $V$  is number of nodes.  $E$  is number of edges.

$\mu$  is weight of nodes.  $v$  is weight of edges.

### 10.5.1 Some information can be used in constructing the graph related to a finger print like:

- Centre of gravity of regions.
- The direction related to the elements of the various regions.
- The area of all the regions.
- The distance between centres of gravity.
- The perimeter of regions.

### 10.5.2 Weight-age to Nodes and Edges

$$W_n = \text{Area}(R_i)$$

Where

- $i = 1, 2, 3 \dots n$ .
- $W_n$  is the weight of nodes.
- $R_i$  is the specified region in block directional image.

$$W_e = (\text{Adj} - p) \times (\text{Node} - d) \times (\text{Diff} - v)$$

Where

- $\text{Adj}-p$  is the boundary of two adjacent regions linking with an edge.
- $\text{Node}-d$  is the distance difference between nodes that links by an edge
- $\text{Diff}-v$  is the phase difference or direction difference between two regions of block directional image.



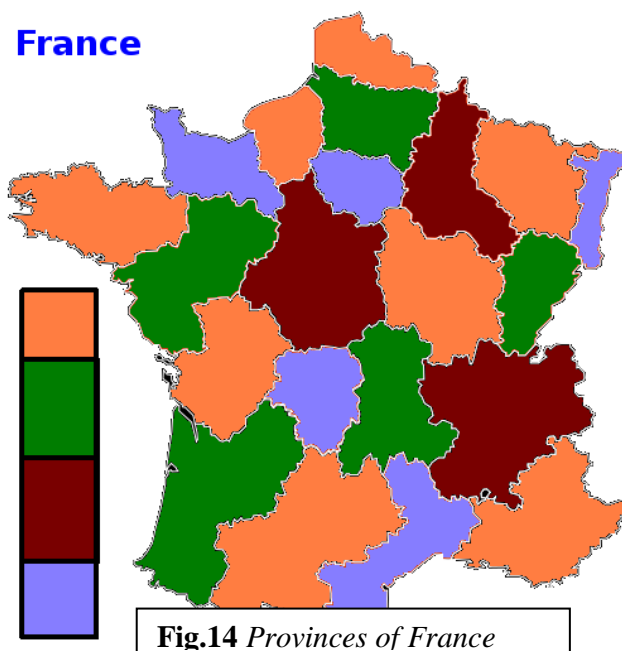
# 11 Graph Theory Models In Security

## Graph theory :

A graph is a simple geometric structure made up of vertices and lines. The lines may be directed arcs or undirected edges, each linking a pair of vertices. Amongst other fields, graph theory as applied to mapping has proved to be useful in Planning Wireless communication networks.

### 11.1 The Four-Color Graph Theorem :

The famous four-color theorem states that for any map, such as that of the contiguous (touching) provinces of France below, one needs only up to four colors to color them such that no two adjacent provinces of a common boundary have the same color. With the aid of computers, mathematicians have been able to prove that this applies for all maps irrespective of the boarder or surface shape



#### 11.1.1 Applying of the four color theorem in wireless a cell tower placement plan.

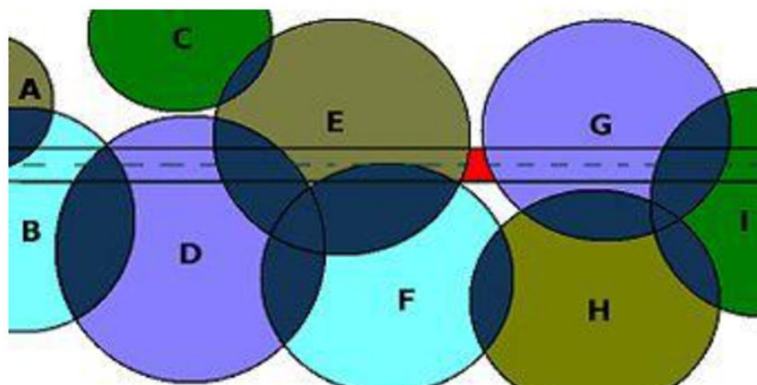
Consider the cell tower placement map shown above, where each cell tower broadcast channel is likened to a color, and channel–

colors are limited to four, the task of finding where to economically position broadcast towers for maximum coverage is equitable to the four-color map problem.

The two challenges are:

1. Elimination of the no-coverage spots ( marked red in the diagram below )
2. Allocation of a different channel in the spots where channel overlap occurs (marked in blue). In analogy, *colors* must be different, so that cell phone signals are *handed off* to a different channel.

Each cell region therefore uses one control tower with a specific channel and the region or control tower adjacent to it will use another tower and another channel. It is not hard to see how by using 4 channels, a node coloring algorithm can be used to efficiently plan towers and channels in a mobile network, a very popular method in use by mobile service providers today



**Fig.15.** Cellular Mobile tower placement map

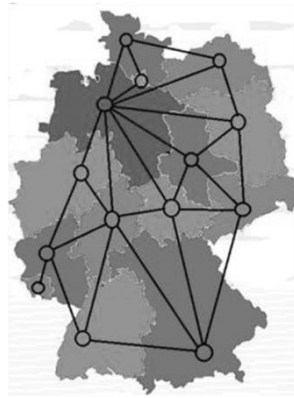
### 11.1.2 Node Coloring Theorem

As can be seen in the map below, borders wander making it a difficult problem to analyze a map. Instead of using a sophisticated map with many wandering boundaries, it becomes a simpler problem if we use node coloring. If two nodes are connected by a line, then they can't be the same color. Wireless Service providers employ node coloring to make an extremely complex network map much more manageable.

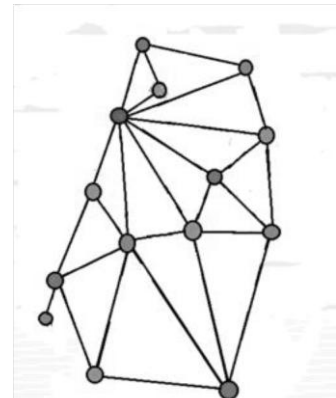




**Fig.16(a)** A Map with complex wandering

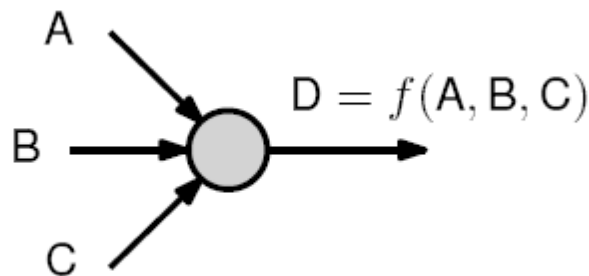


**Fig16(b).** The simplified network version of the map derived by node coloring



## 12 Network Coding

Network coding is another technique where graph theory finds application in mobile communication networks. In a traditional network, nodes can only replicate or forward incoming packets. Using network coding, however, nodes are able to algebraically combine received packets to create new packets



**Fig.17.** Node replication of forward incoming

Network coding opens up new possibilities in the fields of networking. Such would include:

Wireless multi-hop networks:

- ☐ Wireless mesh networks
- ☐ Wireless sensor networks
- ☐ Mobile ad-hoc networks
- ☐ Cellular relay networks

## Peer-to-peer file distribution

- Peer-to-peer streaming
- Distributed storage

### Application of network coding in a content distribution scenario

For this application, the following assumptions are made:

1. The network is a multicast system where all destinations wish to receive similar information from the source.
2. That all Links have a unit capacity of a single packet per time slot
3. That the links be directed such that traffic can only flow in one direction.

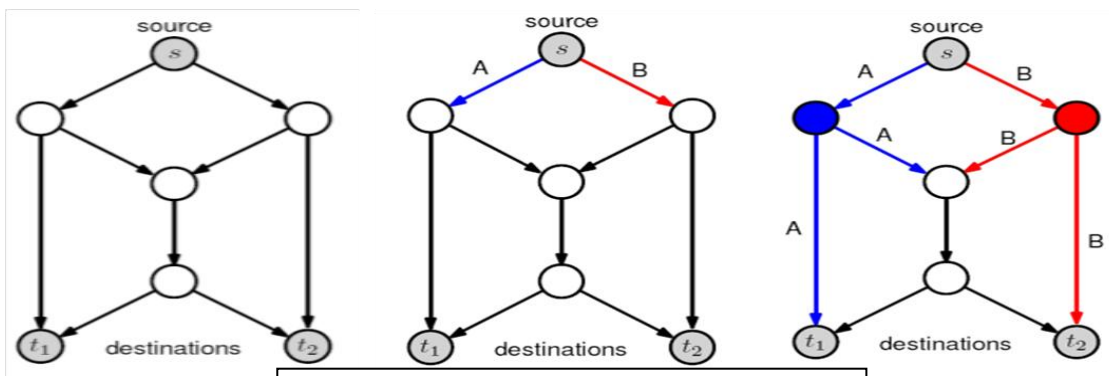


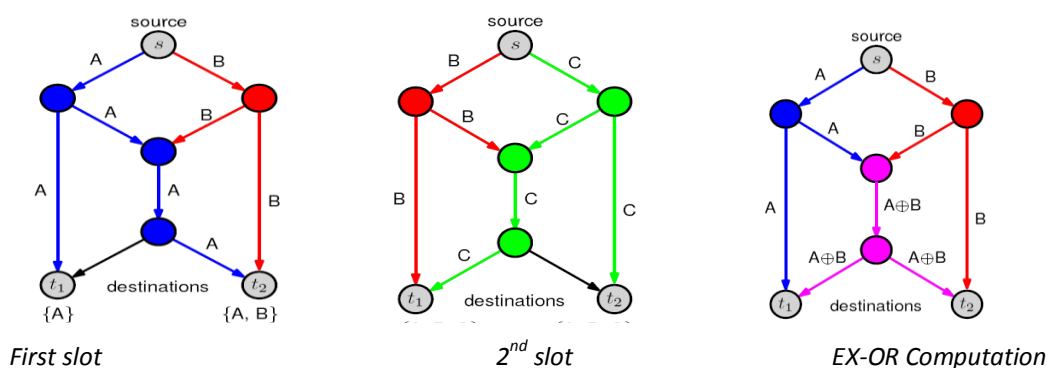
Fig.18 Content distribution scenario

#### 1st time slot

Following the first time slot: Destination  $t_1$  will have received information traffic A whereas Destination  $t_2$  will have received both traffic A and traffic B as shown below.

#### 2nd time slot

In the second time slot: Both Destination  $t_1$  and  $t_2$  will have received traffic A and traffic B and C as shown below

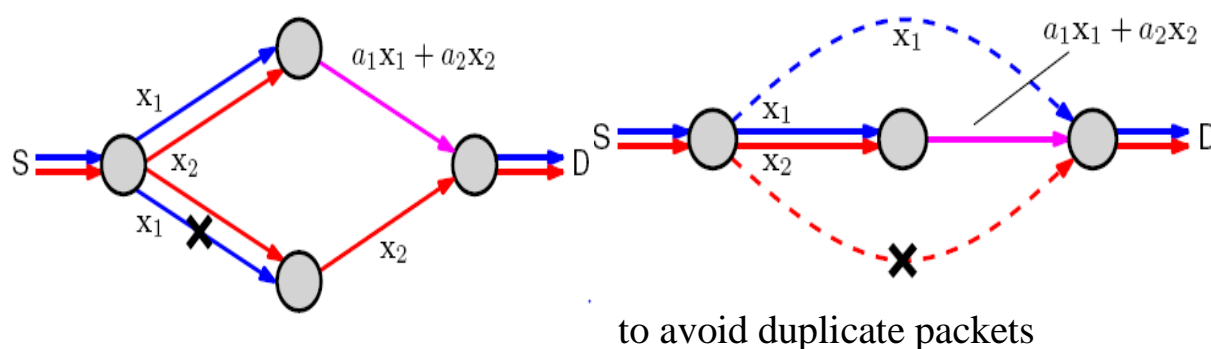


Finally, when Destination t1 receives A and A(EX-OR)B, it will be able to compute B by  $B = A \text{ (EX-OR) } \{ A \text{ (EX-OR) } B \}$

Likewise, when Destination t2 receives B and A (EX-OR) B, it will be able to compute A by:  $A = \{ A \text{ (EX-OR) } B \} \text{ (EX-OR) } B$  as shown below

### Network Coding application in Opportunistic Routing

Opportunistic routing is a technique that makes use of multiple paths in a network to obtain diversity. Network coding can be applied in such a case (fig 9) to coordinate transmissions in order



**Fig 19.** Shows how network coding simply helps us determine how many packets should be sent by each node

## 13 Chemical Graph Theory

Chemical graph theory (CGT) is a branch of mathematical chemistry which deals with the nontrivial applications of graph theory to solve molecular problems. In general, a graph is used to represent a molecule by considering the atoms as the vertices of the graph and the molecular bonds as the edges. Then, the main goal of CGT is to use algebraic invariants to reduce the topological structure of a molecule to a single number which characterizes either the energy of the molecule as a whole or its orbitals, its molecular branching, structural fragments, and its electronic structures, among others.

These graph theoretic invariants are expected to correlate with physical observables measures by experiments in a way that theoretical predictions can be used to gain chemical insights even for not yet existing molecules. In this brief review we shall present a selection of results in some of the most relevant areas of CGT.

## 13.1 DEFINITIONS

**D1:** A molecular graph  $G = (V; E)$  is a simple graph having  $n = |V|$  nodes and  $m = |E|$  edges. The nodes  $v_i \in V$  represent non-hydrogen atoms and the edges  $(v_i; v_j) \in E$  represent covalent bonds between the corresponding atoms. In particular, Hydrocarbons are formed only by carbon and hydrogen atoms and their molecular graphs represent the carbon skeleton of the molecule.

**D2:** An alternant conjugated hydrocarbon is a hydrocarbon with alternant multiple (Double and/or triple) and single bonds, such as the molecular graph is bipartite and the edges of the graph represents  $C = C$  and  $= C - C =$  or  $C \equiv C$  and  $\equiv C - C \equiv$  bonds only

## 13.2 Molecular Energy

- In the Huckel Molecular Orbital (HMO) method for conjugated hydrocarbons the energy of the  $j^{\text{th}}$  molecular orbital of the so-called  $\pi$ -electrons is related to the graph Spectra by

$$\lambda_j = \frac{\alpha - E_{\pi}(j)}{\beta}$$

Where  $\lambda_j$  is an eigenvalue of the adjacency matrix of the hydrogen-depleted graph representing the conjugated hydrocarbon and  $\alpha, \beta$  are empirical parameters [6A]

- The total  $\pi$  (molecular) energy is given by

$$E_{\pi} = \alpha n_e + \beta \sum_{j=1}^n g_j \lambda_j + \beta E$$

Where  $n_e$  is the number of  $\pi$  -electrons in the molecule and  $g_j$  is the occupation number of the  $j^{\text{th}}$  molecular orbital.

- For neutral conjugated systems in their ground state

$$f(n) = \begin{cases} 2 \sum_{j=1}^{n/2} \lambda_j & \text{if } n \text{ is even} \\ 2 \sum_{j=1}^{(n+1)/2} \lambda_j + \lambda_{(j+1)/2} & \text{if } n \text{ is odd} \end{cases}$$

- Let  $G$  be a graph with  $n$  vertices and  $m$  edges. Then,

$$\sqrt{2m + n(n-1) \det(A)^{n/2}} \leq E \leq \sqrt{mn}$$

- Let  $G$  be a graph with  $m$  edges. Then,  $2\sqrt{m} \leq E \leq 2m$
- Let  $G$  be a graph with  $n$  vertices. Then,  $E \geq 2\sqrt{n-1}$ , where the equality holds if  $G$  is the star graph with  $n$  vertices.
- Let  $G$  be a graph with  $n$  vertices. Then,  $E \leq \frac{n}{2}(\sqrt{n} + 1)$ ; Where the equality holds if and only if  $G$  is a strongly regular graph with parameters  $(n, (n+\sqrt{n})/2, (n+2\sqrt{n})/4, (n+2\sqrt{n})/4)$ .

- Let  $G$  be a bipartite graph with  $n$  vertices and  $m$  edges. Then

$$E \leq \frac{4m}{n} + \sqrt{(n-2)(2m - \frac{8m^2}{n^2})};$$

- For all sufficiently large  $n$ , there is a graph  $G$  of order  $n$  such that

$$E \geq \frac{n}{2}(\sqrt{n} - n^{\frac{1}{10}})$$

### 13.3 Graph Nullity and Zero-Energy States

The nullity of a (molecular) graph, denoted by  $\eta = \eta(G)$ , is the algebraic multiplicity of the number zero in the spectrum of the adjacency matrix of the (molecular) graph.

An alternant unsaturated conjugated hydrocarbon with  $\eta = 0$  is predicted to have a stable, closed-shell, electron configuration. Otherwise, the respective molecule is predicted to have an unstable, open-shell, electron configuration.

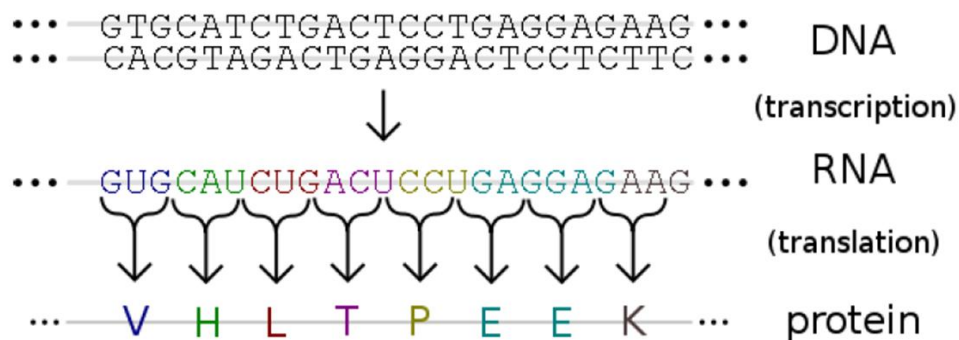
If  $n$  is even, then  $\eta$  is either zero or it is an even positive integer.

## 14 Recent Applications Of Graph Theory In Molecular Biology

### 14.1 New applications in molecular biology

- Graphs as Secondary RNA structures
- Graphs as Amino Acids
- Graphs as Proteins

### DNA to RNA to Protein



Amino Acid 3	letter 1	letter Polarity	Amino Acid 3
Alanine	Ala	A	nonpolar
Arginine	Arg	R	Polar
Asparagine	Asp	N	Polar
Cysteine	Cys	C	Polar
Glutamic acid	Glu	E	nonpolar
Glycine	Gln	Q	Polar
Histidine	His	H	Polar
Isoleucine	Ile	I	nonpolar
Leucine	Leu	L	Polar
Lysine	Lys	K	nonpolar

## ➤ **Human Genome Project**

- How many genes are in the Human Genome?
- Majority of biologists believed a lot more than
- 50,000
- Rice genome contains about 50,000 genes
- Big surprise, humans only have about half that many!

## ➤ **Major changes in the science of molecular biology**

- The number of genes in the genome was just the first of many surprises!
- The belief that the sole function of RNA is to carry the "code" of instructions for a protein is not true
- It is now known that more than half of the RNA molecules that have been discovered are "non-coding"

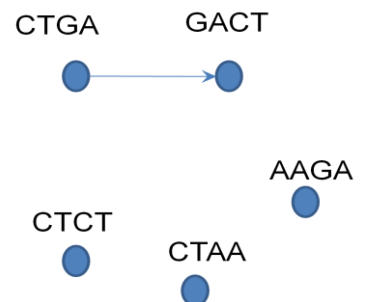
## ➤ **Systems Biology**

- The two surprising discoveries are related
- It is not how many genes; it is how they are regulated that is important!
- Now everyone is trying to understand regulatory networks - Systems Biology

## **14.2A new field has recently emerged called *bioinformatics* – application of IT and CS to molecular biology.**

### **DNA sequencing -**

- Constructing phylogenetic (evolutionary) trees
- Competition graphs.
- Modeling ecological landscapes
- Understanding the dynamics of disease propagation
- Investigating the roles of genes and proteins,
- Etc.



- The study of RNA molecules is at the forefront of the field of Systems Biology

- ## Biology – phylogenetic tree



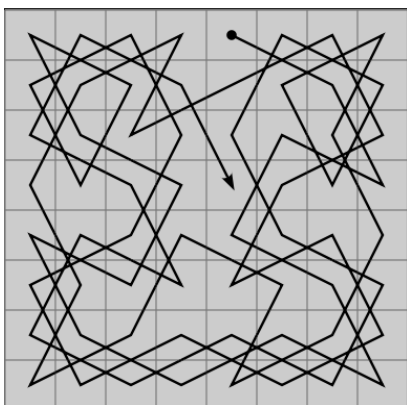


### **Example: studies on bird population**

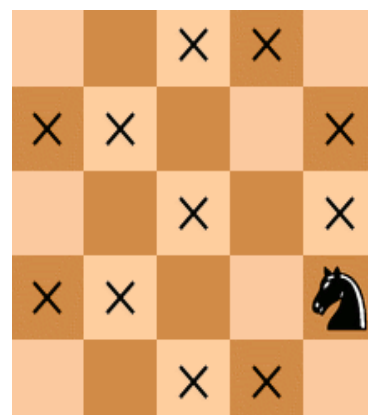
- Ecological landscapes can be modeled using graphs.
- V- represents habitat patches or roosts for birds,  
E - represent movement between the patches.
- Graphs help study the structural organization of a landscape, importance of certain nodes, degree of connectivity between them.
- These properties affect the spread of disease, the vulnerability to disturbance of the landscape, and other issues related to conservation

## **15 Knight's Tour**

- A **knight's tour** is a sequence of moves of a knight on a chessboard such that the knight visits every square only once. If the knight ends on a square that is one knight's move from the beginning square (so that it could tour the board again immediately, following the same path), the tour is *closed*, otherwise it is *open*.
- The **knight's tour problem** is the mathematical problem of finding a knight's tour. Creating a program to find a knight's tour is a common problem given to computer science students.<sup>[1]</sup> Variations of the knight's tour problem involve chessboards of different sizes than the usual  $8 \times 8$ , as well as irregular (non-rectangular) boards.



An open knight's tour of a chessboard



- The knight's tour problem is an instance of the more general Hamiltonian path problem in graph theory. The problem of finding a closed knight's tour is similarly an instance of the Hamiltonian cycle problem. Unlike the general Hamiltonian path problem, the knight's tour problem can be solved in linear time.
- There are quite a number of ways to find a knight's tour on a given board with a computer. Some of these methods are algorithms while others are heuristics.

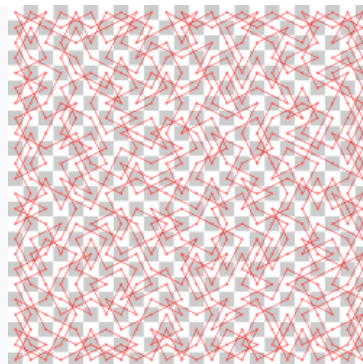
### 1.1.1 Brute-force algorithms

A brute-force search for a knight's tour is impractical on all but the smallest boards;<sup>[10]</sup> for example, on an  $8 \times 8$  board there are approximately  $4 \times 10^{51}$  possible move sequences,<sup>[11]</sup> and it is well beyond the capacity of modern computers (or networks of computers) to perform operations on such a large set. However, the size of this number gives a misleading impression of the difficulty of the problem, which can be solved "by using human insight and ingenuity ... without much difficulty."<sup>[10]</sup>

### 1.1.2 Divide and conquer algorithms

By dividing the board into smaller pieces, constructing tours on each piece, and patching the pieces together, one can construct tours on most rectangular boards in linear time - that is, in a time proportional to the number of squares on the board.<sup>[5][12]</sup>

### 1.1.3 Neural network solutions



Closed knight's tour on a  $24 \times 24$  board solved by a neural network.

The Knight's Tour problem also lends itself to being solved by a neural network implementation.<sup>[13]</sup> The network is set up such that every legal knight's move is represented by a neuron, and each neuron is initialized randomly to be either "active" or "inactive" (output of 1 or 0), with 1 implying that the neuron is part of the final solution. Each neuron also has a state function (described below) which is initialized to 0.

When the network is allowed to run, each neuron can change its state and output based on the states and outputs of its neighbors (those exactly one knight's move away) according to the following transition rules:

$$U_{t+1}(N_{i,j}) = U_t(N_{i,j}) + 2 - \sum_{N \in G(N_{i,j})} V_t(N)$$

$$V_{t+1}(N_{i,j}) = \begin{cases} 1 & \text{if } U_{t+1}(N_{i,j}) > 3 \\ 0 & \text{if } U_{t+1}(N_{i,j}) < 0 \\ V_t(N_{i,j}) & \text{otherwise,} \end{cases}$$

where  $t$  represents discrete intervals of time,  $U(N_{i,j})$  is the state of the neuron connecting square  $i$  to square  $j$ ,  $V(N_{i,j})$  is the output of the neuron from  $i$  to square  $j$  and  $G(N_{i,j})$  is the set of neighbours of the neuron.

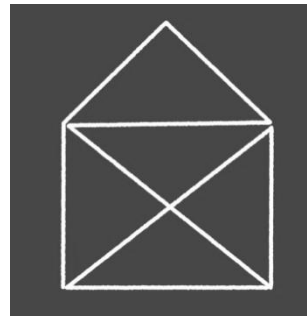
- Although divergent cases are possible, the network should eventually converge, which occurs when no neuron changes its state from time  $t$  to  $t+1$ . When the network converges, either the network encodes a knight's tour or a series of two or more independent circuits within the same board.

## 16 Common Problems

### 16.1 House of Santa Claus:

The house in the picture can be represented as a graph, where each vertex is a node. Try to draw a the house by yourself following this rules:

- You have to draw a house in one line.
- You must not lift your pencil while drawing.
- You must not repeat a line.



### 16.2 Three utilities problem:

The classical mathematical puzzle known as the **three utilities problem**; the **three cottages problem** or sometimes **water, gas and electricity** can be stated as follows:

There are three utilities (three vertices) and three houses (three vertices), and the purpose of this problem is to draw a line from each house to each facility without the lines ever crossing.



Suppose there are three cottages on a plane (or sphere) and each needs to be connected to the gas, water, and electricity companies. Without using a third dimension or sending any of the connections through another company or cottage, is there a way to make all nine connections without any of the lines crossing each other?

The problem is an abstract mathematical puzzle which imposes constraints that would not exist in a practical engineering situation. It is part of the mathematical field of topological graph theory which studies the embedding of graphs on surfaces. In more formal graph-theoretic terms, the problem asks whether the complete bipartite graph  $K_{3,3}$  is planar.<sup>[1]</sup> This graph is often referred to as the **utility graph** in reference to the problem;<sup>[2]</sup> it has also been called the **Thomsen graph**

### 16.3 Seven Bridges of Königsberg:

The **Seven Bridges of Königsberg** is a historically notable problem in mathematics. Its negative resolution by Leonhard Euler in 1736<sup>[16]</sup> laid the foundations of graph theory and prefigured the idea of topology.<sup>[16]</sup>

The city of Königsberg in Prussia (now Kaliningrad, Russia) was set on both sides of the Pregel River, and included two large islands which were connected to each other, or to the two mainland portions of the city, by seven bridges. The problem was to devise a walk through the city that would cross each of those bridges once and only once.

By way of specifying the logical task unambiguously, solutions involving either

1. reaching an island or mainland bank other than via one of the bridges, or
2. accessing any bridge without crossing to its other end

are explicitly unacceptable.

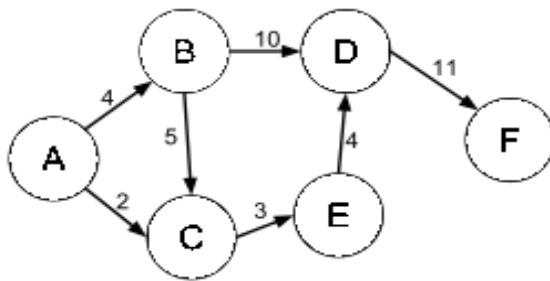
Euler proved that the problem has no solution. The difficulty he faced was the development of a suitable technique of analysis, and of subsequent tests that established this assertion with mathematical rigor.



## 16.4 Shortest path problem:

In graph theory, the **shortest path problem** is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its constituent edges is minimized.

The problem of finding the shortest path between two intersections on a road map (the graph's vertices correspond to intersections and the edges correspond to road segments, each weighted by the length of its road segment) may be modeled by a special case of the shortest path problem in graphs



Shortest path (A, C, E, D, F) between vertices A and F in the weighted directed graph

## 16.5 Travelling Salesman Problem (TSP):

The **travelling salesman problem (TSP)** asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?" It is an NP-hard problem in combinatorial optimization, important in operations research and theoretical computer science.

The travelling purchaser problem and the vehicle routing problem are both generalizations of TSP.

In the theory of computational complexity, the decision version of the TSP (where, given a length  $L$ , the task is to decide whether the graph has any tour shorter than  $L$ ) belongs to the class of NP-complete problems. Thus, it is possible that the worst-case running time for any algorithm for the TSP increases superpolynomially (but no more than exponentially) with the number of cities.

The problem was first formulated in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, a large number of heuristics and exact algorithms are known, so that some instances with tens of thousands of cities can be solved completely and even problems with millions of cities can be approximated within a small fraction of 1%.

The TSP has several applications even in its purest formulation, such as planning, logistics, and the manufacture of microchips. Slightly modified, it appears as a sub-problem in many areas, such as DNA sequencing. In these applications, the concept *city* represents, for example, customers, soldering points, or DNA fragments, and the concept *distance* represents travelling times or cost, or a similarity measure between DNA fragments. The TSP also appears in astronomy, as astronomers observing many sources will want to minimize the time spent moving the telescope between the sources. In many applications, additional constraints such as limited resources or time windows may be imposed

## **Bibliography**

[4A]:- Harary & Sumner (1980).

[4B]:- Simion (1991).

[4C]:- Kim & Pearl (1983).

[6A]:- C. A. Coulson, B. O'Leary, and R. B. Mallion, Huckel theory for organic chemists. Academic Press, London, 1978.

[6B]:- Rosen, Kenneth H. (1999), Handbook of Discrete and Combinatorial Mathematics, Discrete Mathematics and Its Applications, CRC Press, p. 515

[5]:- Cull, P.; De Curtins, J. (1978)

[10]:- Simon, Dan (2013), Evolutionary Optimization Algorithms, John Wiley & Sons, pp. 449– 450, ISBN 9781118659502, The knight's tour problem is a classic combinatorial optimization problem

[11]:- "*Enumerating the Knight's Tour*

[12]:- arberry, Ian (1997). "An Efficient Algorithm for the Knight's Tour Problem" . *Discrete Applied Mathematics*

[13]:- Y. Takefuji, K. C. Lee. "Neural network computing for knight's tour problems." *Neurocomputing*, 4(5):249–254, 1992.

[16]:- Euler, Leonhard (1736). "Solutio problematis ad geometriam situs pertinentis". *Comment. Acad. Sci. U. Petrop* 8. Shields, Rob (December 2012). "Cultural Topology: The Seven Bridges of Königsburg 1736



Thank you



manikanta4200043@gmail.com

manikanta4200043@gmail.com