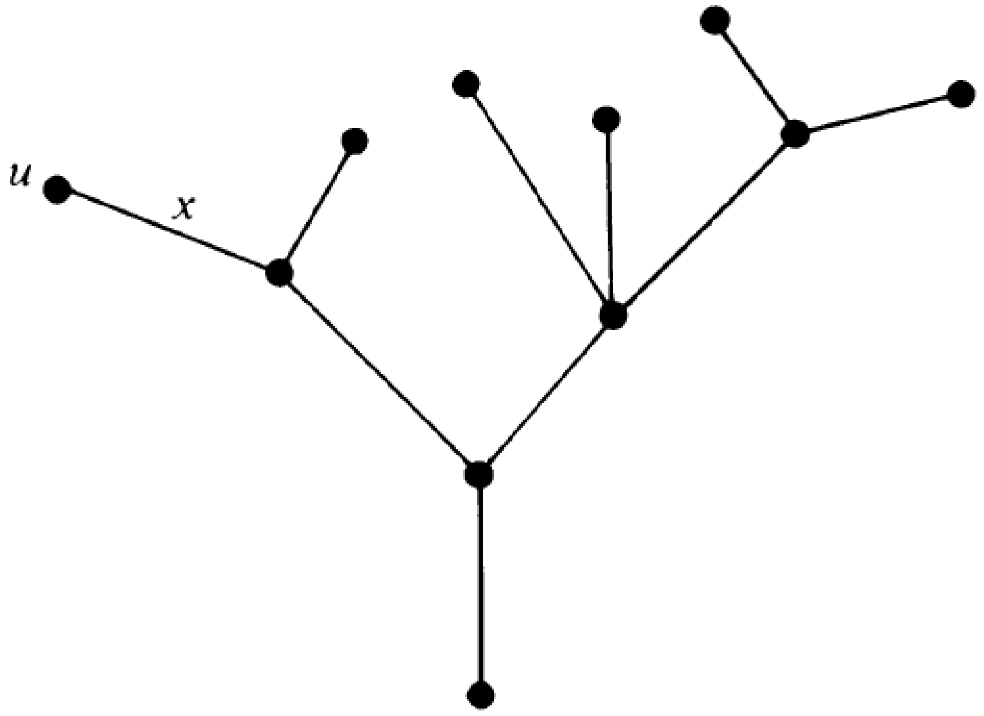
Module 3

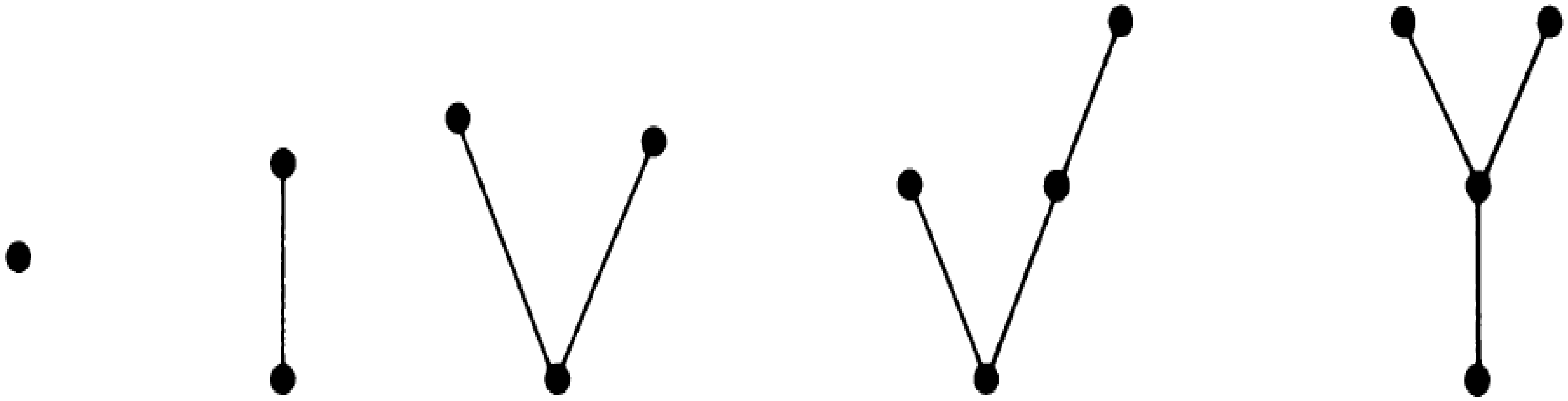
Trees

Trees

- It is a connected graph without any circuits
- Parallel edges and self loop are not possible



Tree.



Trees with one, two, three, and four vertices.

Properties of Trees

- **Theorem: 1**

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 at least one path between very pair of vertices in T. Suppose between 2 vertices a and b of T there are 2 distinct paths. The union of these paths will contain a circuit and T cannot have a circuit.

Theorem – 2

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

Proof: Existence of a path between every pair of vertices assures that G is a connected graph. A circuit in a graph implies that there is at least one pair of vertices a, b such that there exist two distinct paths between a and b . Since G has one and only one path between every pair of vertices, G have no circuit. Therefore G is a tree.

Theorem - 3

A tree with n vertices has n-1 edges

Proof:

Let us now consider a tree T with n vertices. In T let e_k be an edge with end vertices v_i and v_j. According to Theorem 3-1, there is no other path between v_i and v_j except e_k. Furthermore, $T - e_k$ consists of exactly two components, and since there were no circuits in T to begin with, each of these components is a tree. Both these trees, t_1 and t_2, have fewer than n vertices each, and therefore, by the induction hypothesis, each contains one less edge than the number of vertices in it. Thus $T - e_k$ consists of $n - 2$ edges (and n vertices). Hence T has exactly $n - 1$ edges.

Theorem – 4

Any connected graph with n vertices and n-1 edges is a tree.

Proof: ???

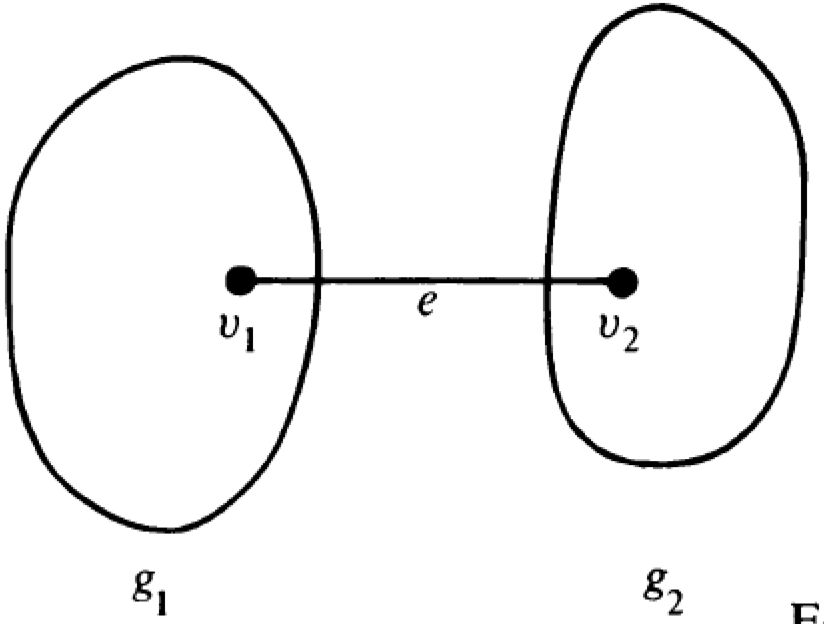
Minimally Connected Graph

- A connected graph is said to be minimally connected, if removal of any one edge from it disconnects the graph.
- It cannot have circuits
- So Minimally Connected Graph will a tree.
- Or, if a connected graph is not Minimally Connected Graph then, there exist an edge e_i such that, $G - e_i$ is connected.

Theorem – 5

A graph is a tree iff it is minimally connected.

Proof : *To interconnect n distinct points, we need n-1 line segments*



Edge e added to $G = g_1 \cup g_2$.

Theorem – 6

A graph G with n vertices, n-1 edges and no circuits is connected.

Proof: *Suppose there exist a circuit-less graph with n vertices and n-1 edges which are disconnected. Then G will consists of two or more circuit less components.*

Let g1 and g2 be two components. Add an edge e between v1 in g1 and v2 in g2 . Since there is no path between v1 and v2 won't create a circuit on adding e. Thus ***G U e*** *is a circuit less connected graph (tree).*

Properties of tree(Summary)

- A graph G with n vertices is called a tree if :

  - ***G is connected and circuit less or,***
  - ***G is connected and has n-1 edges or,***
  - ***G is circuit less and has n-1 edges or***
  - ***There is exactly one path between every pair of vertices in G, or***
  - ***G is minimally connected graph.***

Pendant vertices in a tree

- Pendant vertex is a vertex with degree one.
- In a tree with n vertices and n-1 edges will be present
- Each edge contribute 2 degrees
- So 2(n-1) degrees should be divided among vertices
- Since no vertex can have degree zero, we must have at least 2 vertices of degree one(if n >=2)

Theorem – 7

In any tree(with two or more vertices), there are at least two pendant vertices.

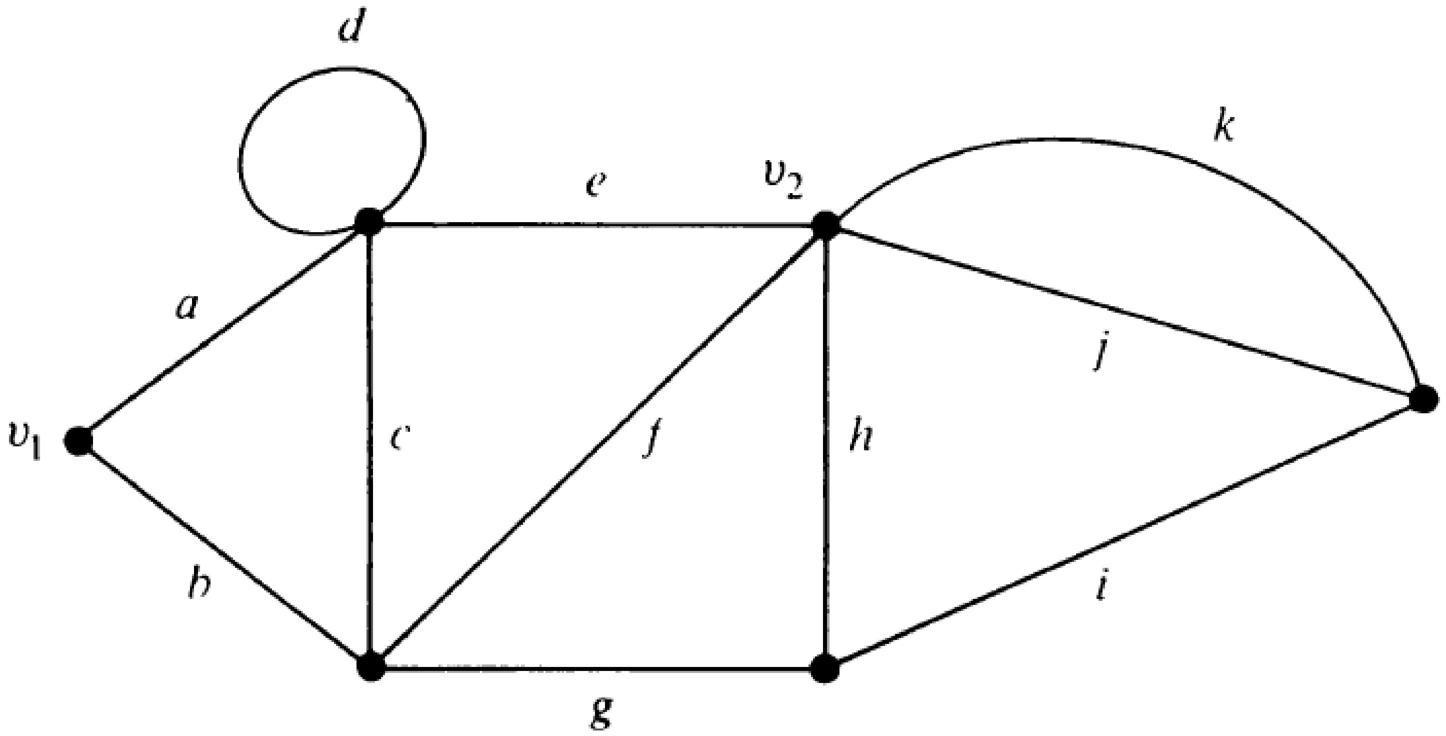
Proof????

Distance and centers in a tree

In a connected graph G, the distance d(v_i , v_j) between two of its vertices v_i and v_j is the length of the shortest path(ie, the number of edges in the shortest path) between them.

Find paths between v1 and v2??????

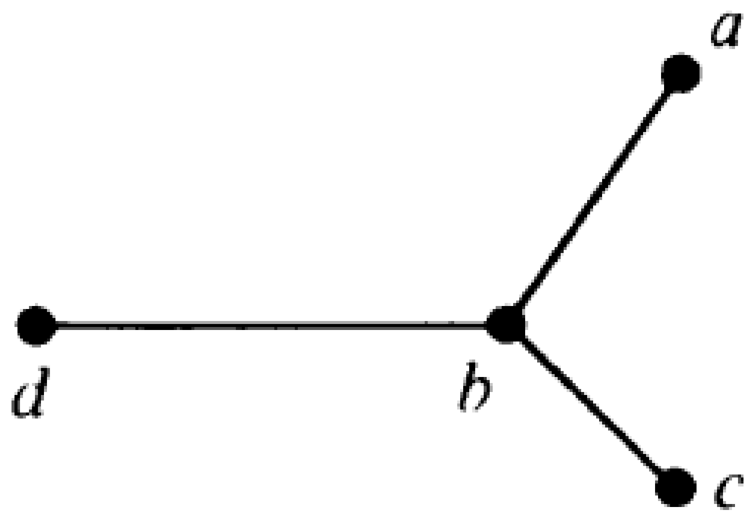
$d(v_1, v_2) = 2$



d(a,b)=?

d(a,c)=?

d(c,b)=?



Metric

1. Nonnegativity: $f(x, y) \geq 0$, and $f(x, y) = 0$ if and only if $x = y$.
2. Symmetry: $f(x, y) = f(y, x)$.
3. Triangle inequality: $f(x, y) \leq f(x, z) + f(z, y)$ for any z.

- A function that satisfies these three conditions is called a metric.

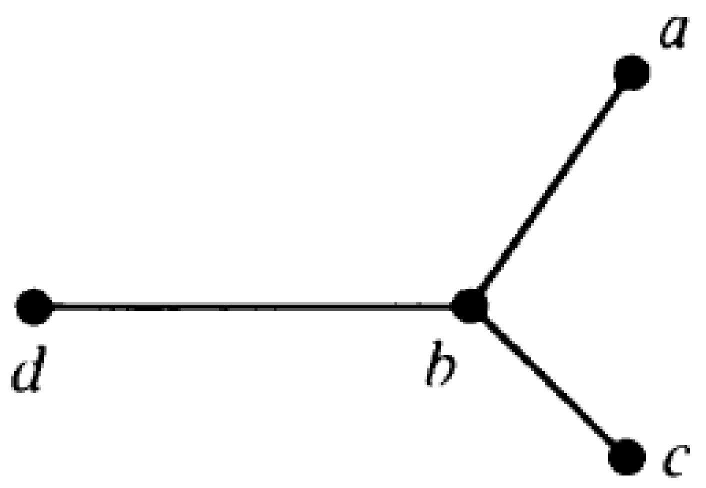
Theorem

The distance between vertices of a connected graph is a metric

- Eccentricity E(v) of a vertex v in graph G is the distance from v to vertex farthest from v in G.
- Also known as separation or associated number.

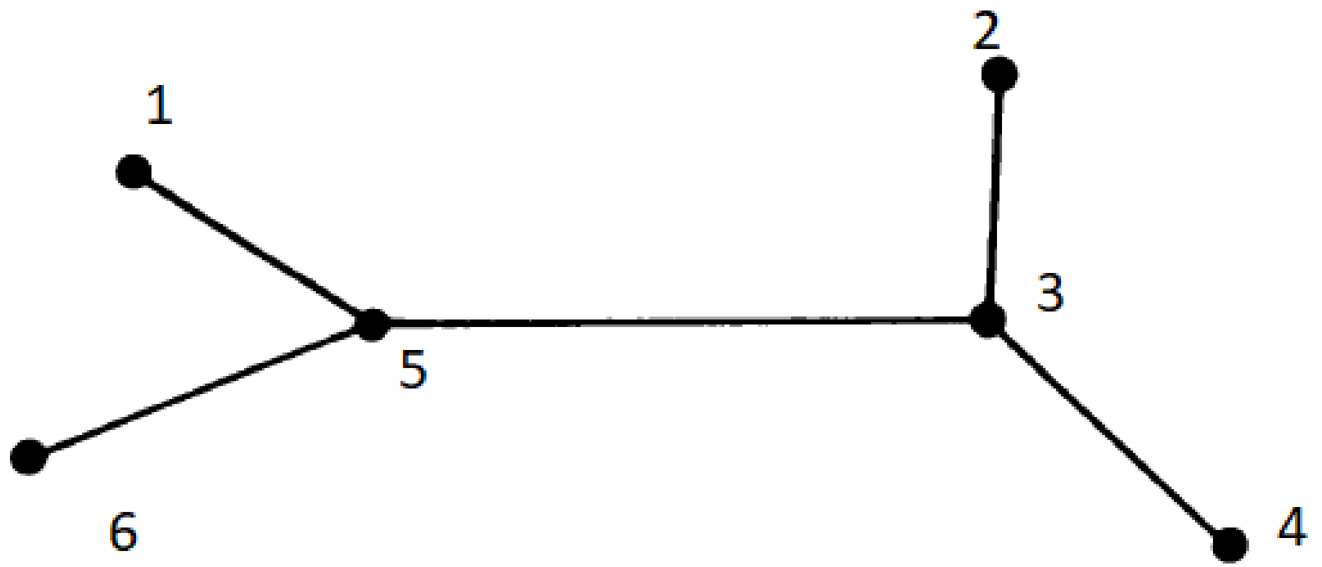
$$E(v) = \max_{v_1 \in G} d(v, v_i).$$

- A vertex with minimum eccentricity in a graph G is called center of graph.
- E(a)=2
- E(b)=1
- E(c) =2
- E(d)=2



- So b is the center of graph

Find eccentricity of each vertices



- This tree has 2 vertices having same minimum eccentricity. Hence this tree has two centers. Thus this type of tree are called bicenters

Theorem

Every tree has either one or two centers

Proof: *max d(v , v_i) from a given vertex v to any other vertex v_i occurs only when v_i is a pendant vertex.*

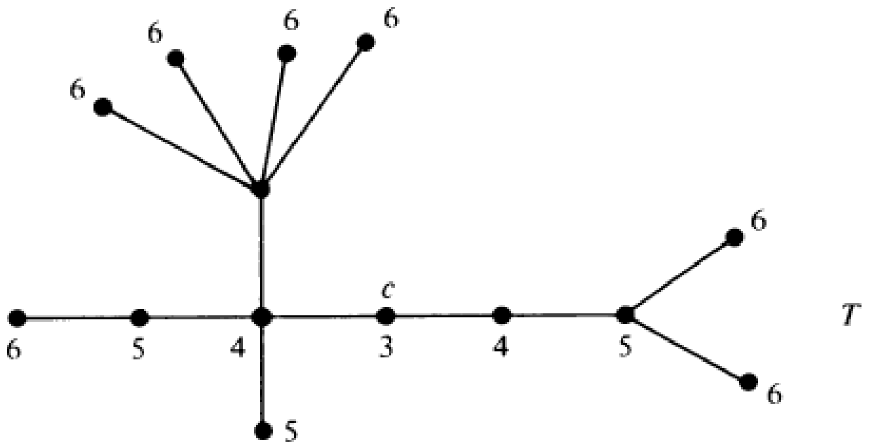
Let T be a tree having more than 2 vertices. A tree T will have two or more pendant vertices.

Delete all pendant vertices from T. Thus form T'. T' still a tree.

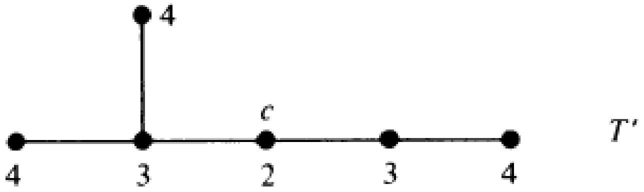
Find eccentricities of every vertices in T'. And find center of T'.

Centers of T still remain as centers of T'.

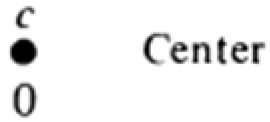
Continue removal of pendant vertices until an edge or vertex remain. Thus theorem proved.



(a)



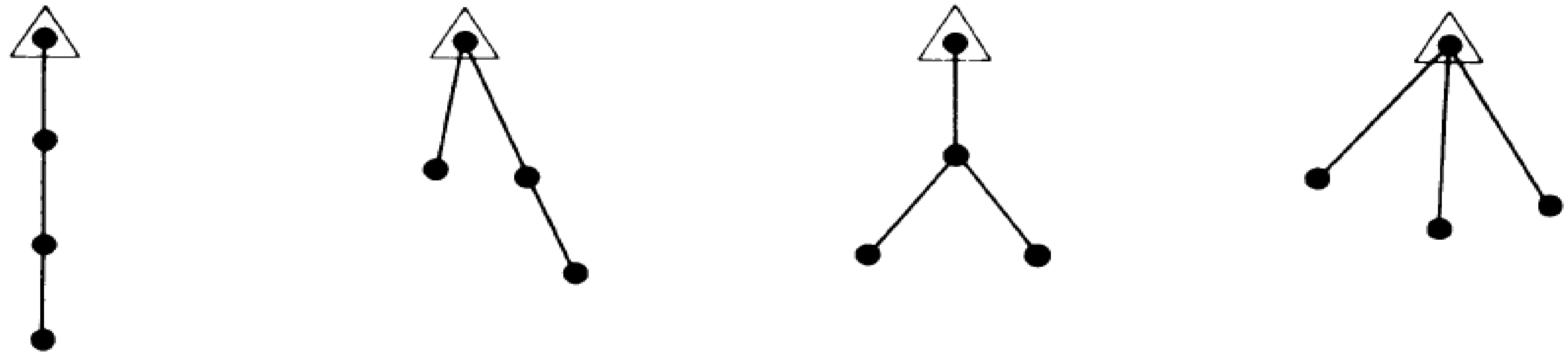
(b)



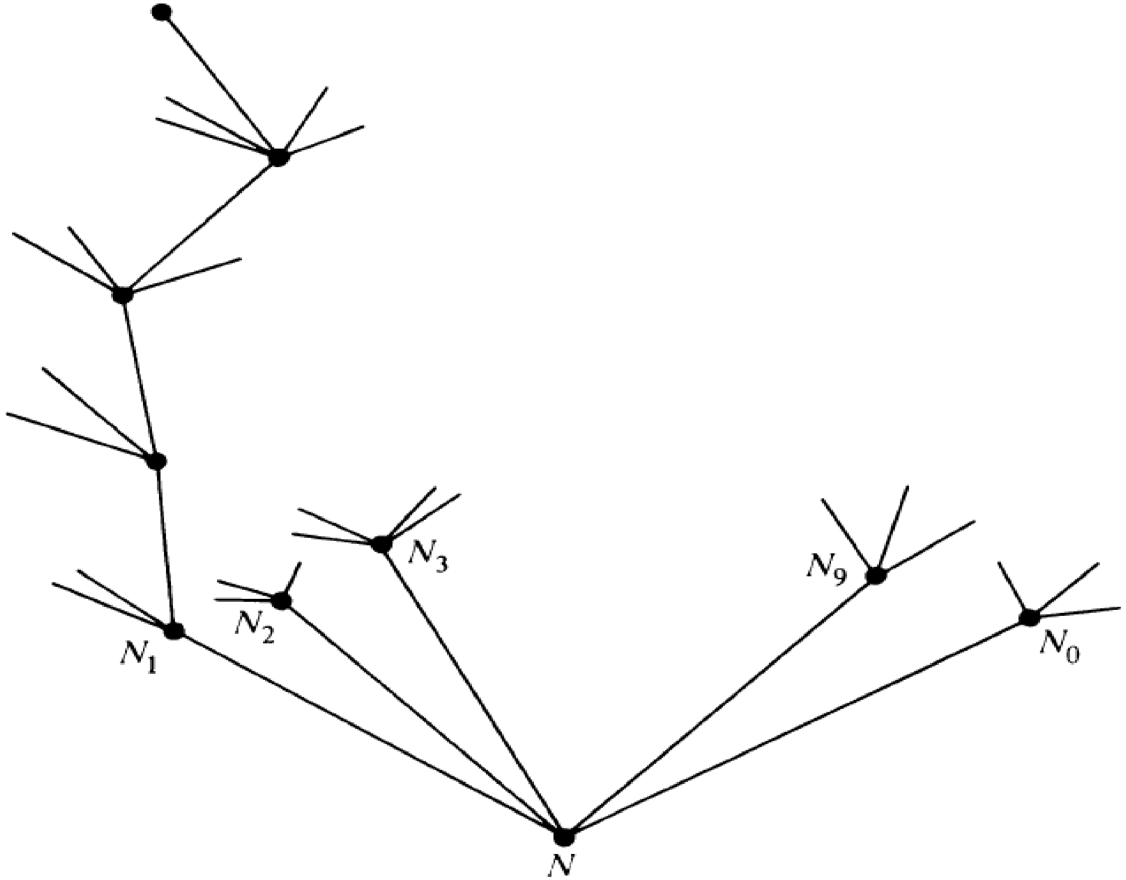
(d)

Rooted Tree

- A tree in which a vertex (root) is distinguished from all other vertices is called rooted tree.

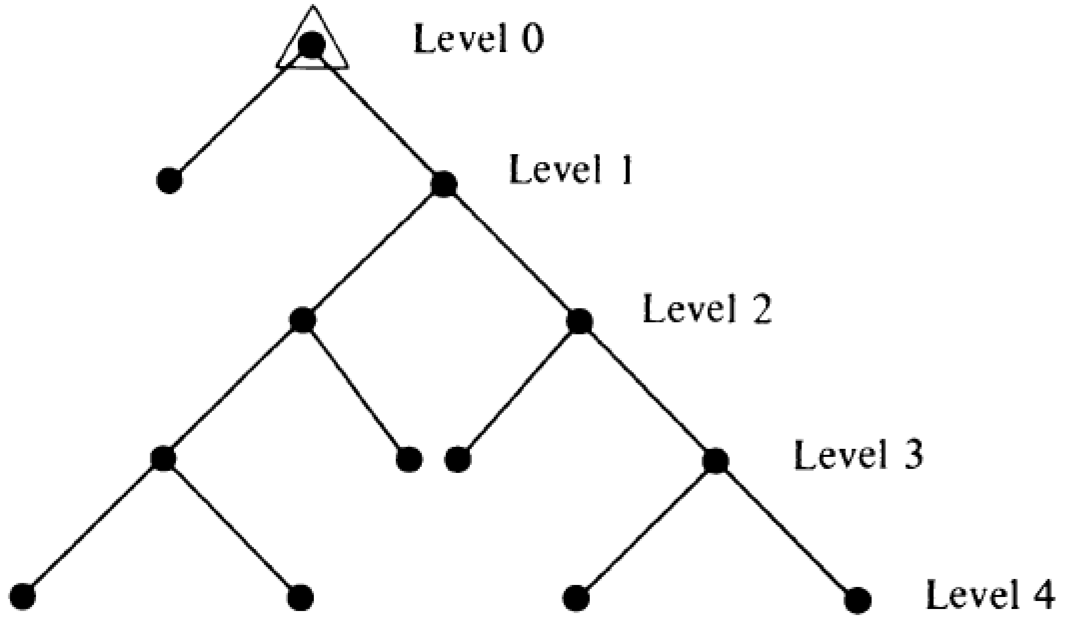


Rooted trees with four vertices.


N_3
N_2
N_1
N_9
N_0
N

Binary Tree

- Special kind of rooted tree
- It is a tree in which there is exactly one vertex of degree two, and each of the remaining vertices is of degree one or three.



A 13-vertex, 4-level binary tree.

Properties of binary tree

1) The number of vertices n in a binary tree is always odd

- Because there is exactly one vertex of even degree and remaining n-1 vertices are of odd degrees. By theorem , the number of odd degree vertices are even, ie, n-1 is even. Hence n is odd.

2)Given p and n, then total number of edges in T is

$$\tfrac{1}{2}[p+3(n-p-1)+2]=n-1$$

- p –> number of pendent vertices in a binary tree T.
- n -> number of vertices in T

n-p-1 is the number of vertices of degree three. Thus, number of edges in T is

$$\tfrac{1}{2}[p+3(n-p-1)+2]=n-1$$

$$\text{Thus, } p=[n+1]/2$$

- A non pendant vertex in a tree -> internal vertex
- The number of internal vertices is one less than the number of pendant vertices
- In a binary tree, a vertex v_i is said to be in level l_i , if v_i is at a distance l_i from root.
- Root vertex is at level 0 – only one vertex
- In level 1, there can be at most 2 vertices
- In level 2, there can be at most 4 vertices and so on………..

- Maximum number of vertices possible in k level binary tree :

$$2^0 + 2^1 + 2^2 + \cdots + 2^k$$

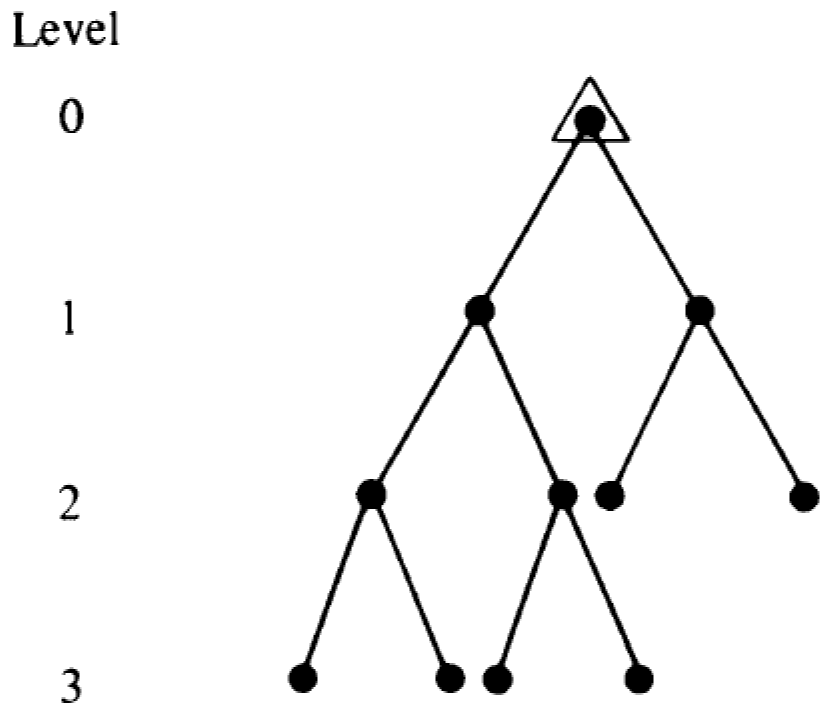
- The minimum possible height of a n – vertex binary tree is

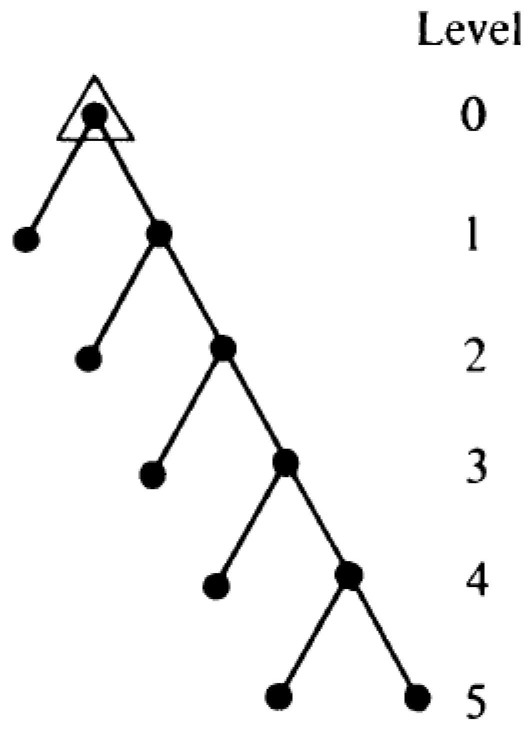
$$\min l_{\max} = \lceil \log_2 (n + 1) - 1 \rceil$$

- The maximum possible height of a n – vertex binary tree is

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

Maximum & minimum height of 11- vertex tree



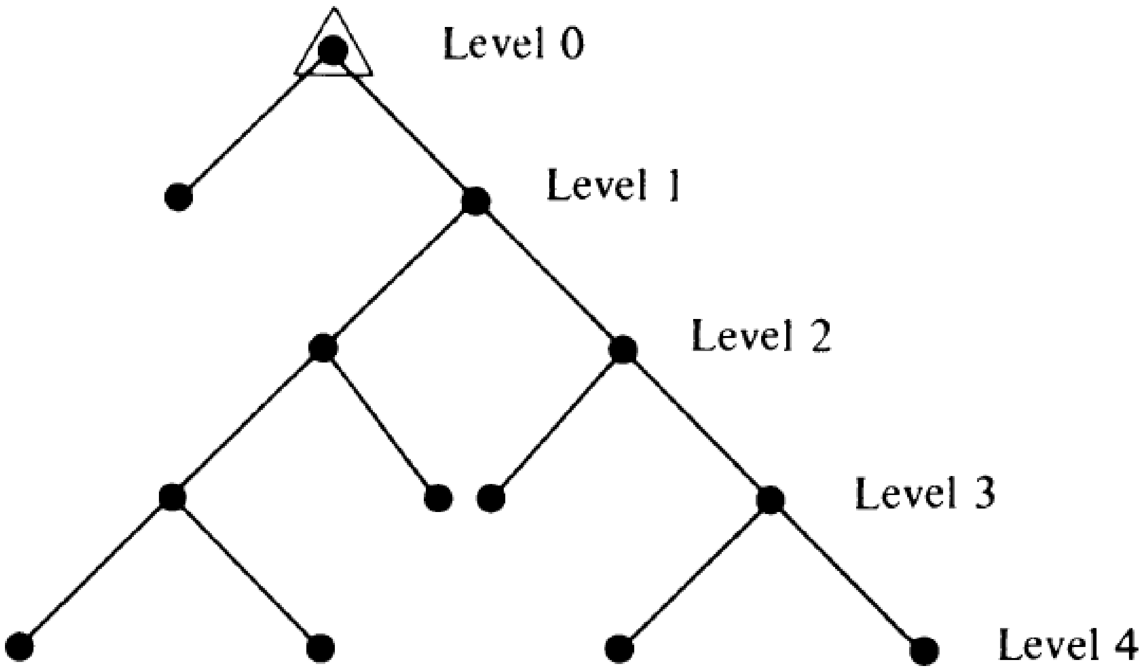


$\min l_{max} = \lceil(\log_2 12) - 1\rceil$

$\max l_{max} = \frac{11 - 1}{2} = 5$

Path length

- Also known external path length
- It is the sum of the path length from the root to all pendant vertices



Weighted path length

- Every pendant vertex v_j of binary tree is associated with a positive real number w_j

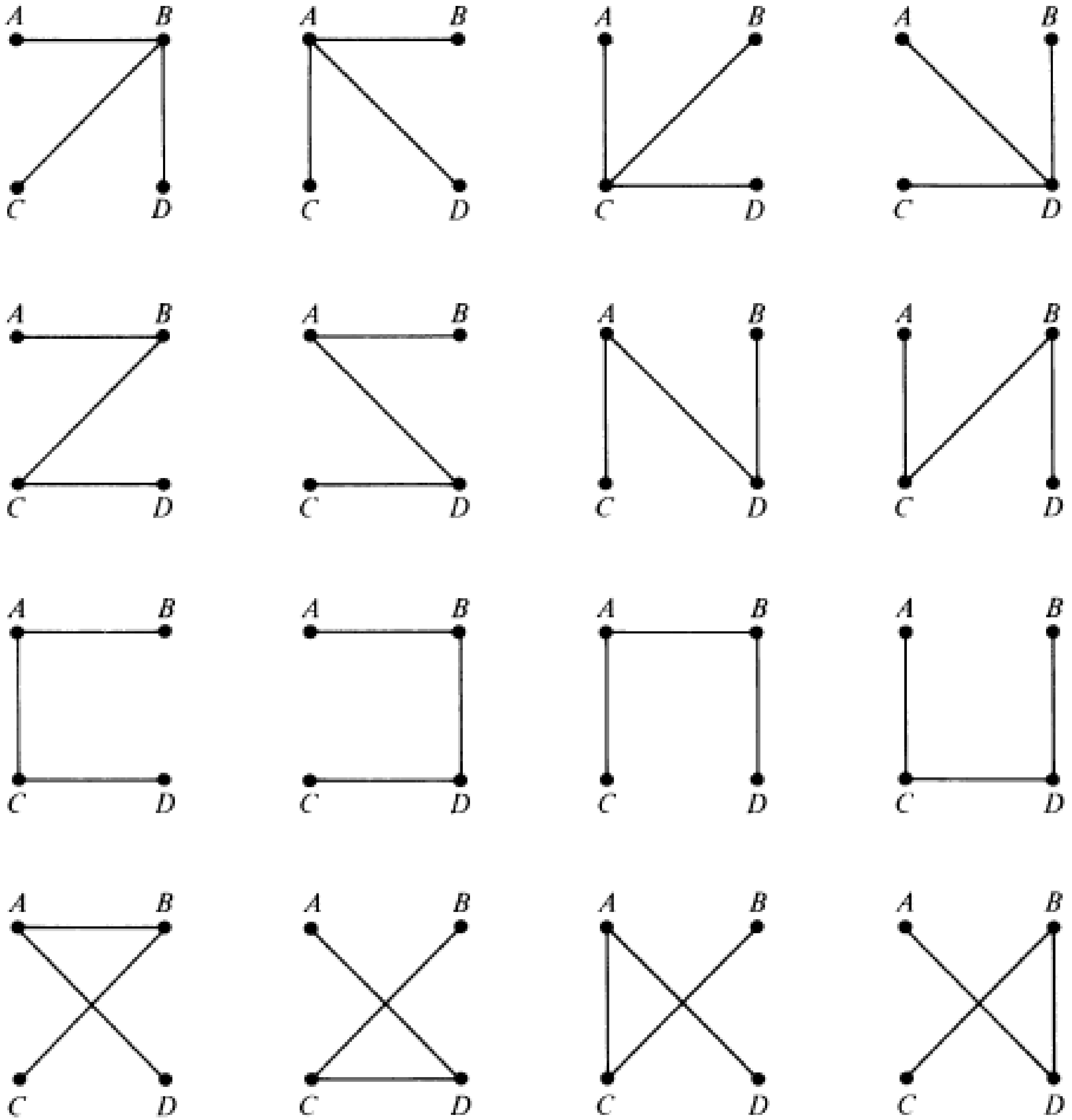
- Weighted path length is given as :

$$\sum w_j l_j$$

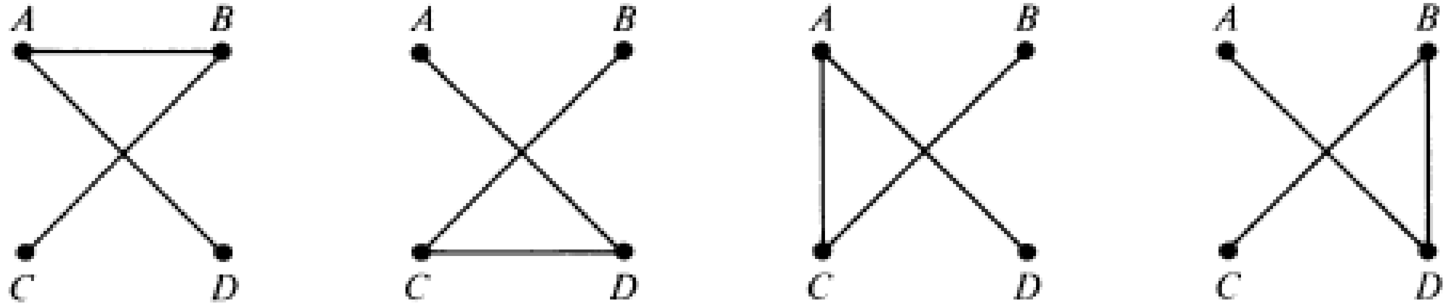
Counting Trees

What is the number of different trees that can be constructed from n distinct (or labelled) of vertices?????????

Consider n=4


A B C D

- Labelling is very important, else below 4 trees will be considered as same.

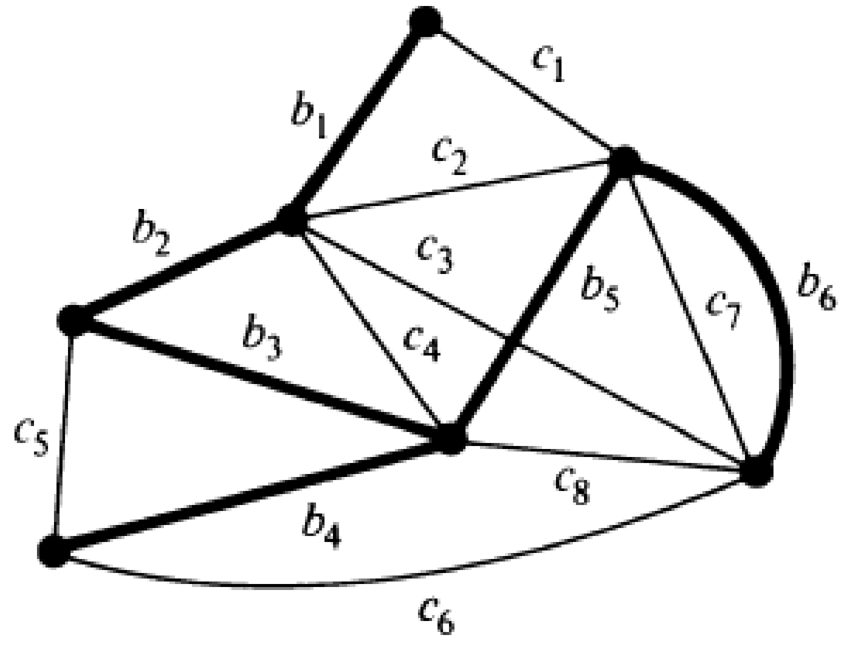


Cayley’s Theorem

The number of labelled trees with n vertices ($n>=2$) is n^{n-2}

Spanning Trees

- A tree T is said to be a spanning tree of a connected graph G, if T is a subgraph of G and T contains all vertices of G.



- It is known as ***skeleton*** of G, since it contains all vertices in G.

- Spanning trees are the largest trees among all trees in G. So spanning trees can be also called ***maximal tree subgraph or maximal tree of G.***

- Spanning tree is normally defined in connected graphs.

- Disconnected graph with k components has ***spanning forest*** consisting of k spanning trees.

Finding spanning tree from G

- If G has no circuit, it is its own spanning tree.

- If G has a single circuit, deleting an edge from circuit will provide spanning tree of G.

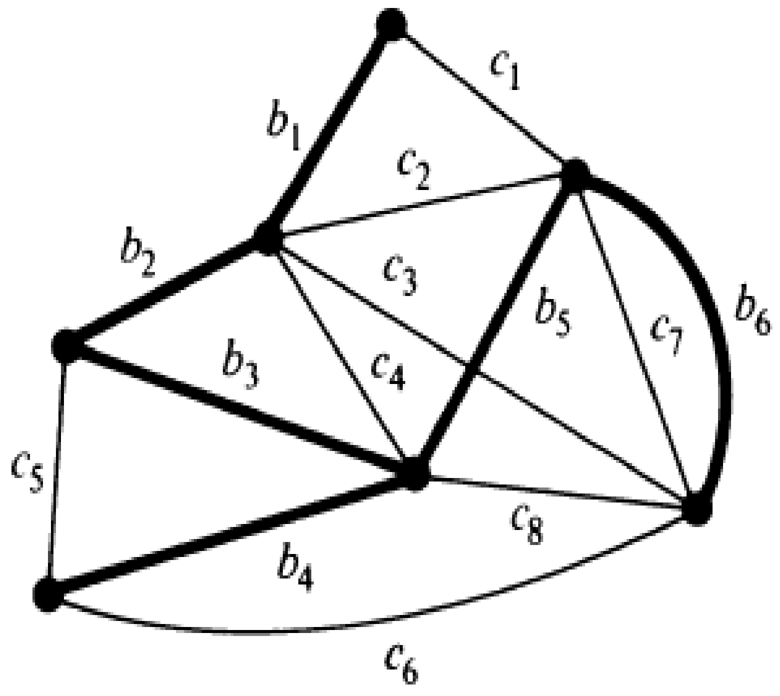
- If G has more circuits, then repeat the operation till a ***circuit - free, connected*** graph that contains all vertices in G is formed.

Theorem

Every connected graph has at least one spanning tree

Proof: Let G be a connected graph. If G has no cycles, then it is its own spanning tree. If G has cycles, then on deleting one edge from each of the cycles, the graph remains connected and cycle free containing all the vertices of G.

- Edge in spanning tree - **branch**
- Edge of G which is not in given spanning tree T – **chord/ link/ tie**



- A connected graph G is a union of T and T’

$$G = T \cup T'$$

where T is spanning tree in G and T’ is the complement of T in G(T’ is the collection of chords)

Theorem

With respect to any of its spanning trees, a connected graph of n vertices and e edges has n-1 branches and e-n+1 chords

Proof: Let G be a connected graph with n vertices and e edges.

Let T be the spanning tree.

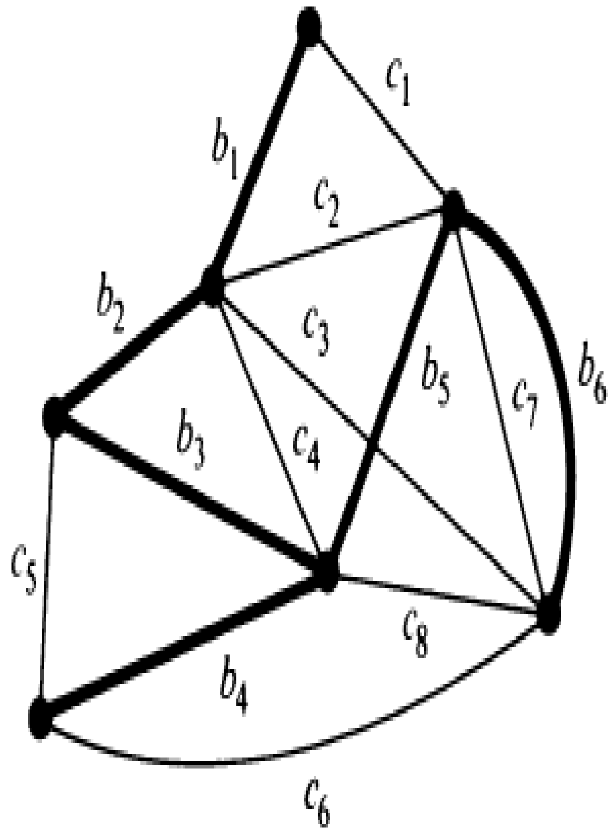
Since T contains all n vertices of G, T has n−1 edges

Thus the number of chords in G is equal to ***e−(n−1) = e−n+1.***

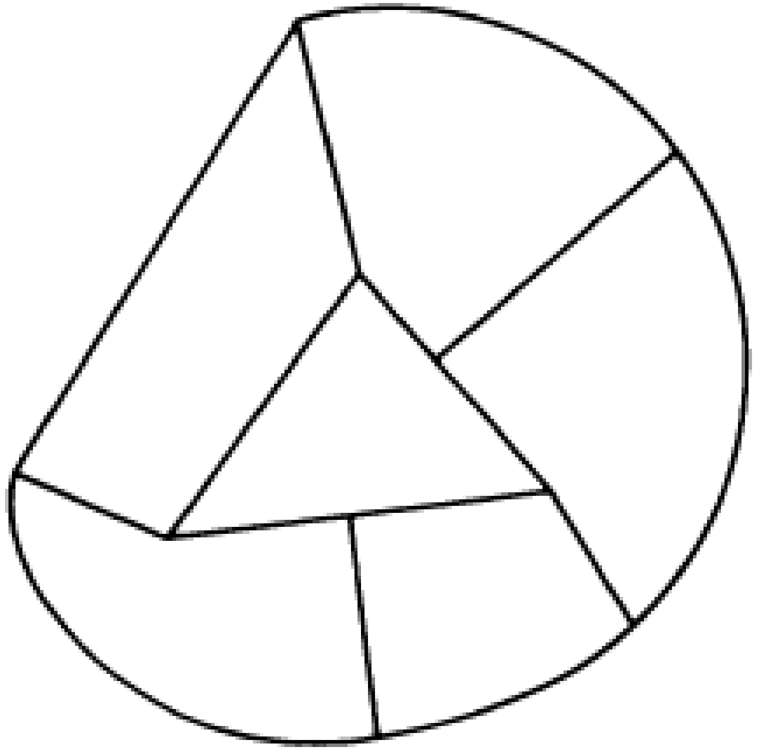
Consider n=7 and e=14

Spanning tree T ={b1, b2, b3, b4, b5, b6}

Six tree branches and eight chords



- An electric network with x elements and y nodes , what is the minimum number of elements we must remove to eliminate circuits ????????



- A graph G consists of
  - n vertices
  - e edges
  - k components
- Relation between n, e & k
  - n >=k –>Since every component in G has at least one vertex
  - The number of edges in a component can be no less than the number of vertices in that component minus one.(e>=n-1)

$$\text{rank} \qquad r = n - k,$$

$$\text{nullity} \qquad \mu = e - n + k.$$

- Rank of G = number of branches in any spanning tree/forest of G
- Nullity of G = number of chords in G
- Rank + Nullity = number of edges in G

- Nullity also referred as ***cyclomatic number or first Betti number***

Theorem

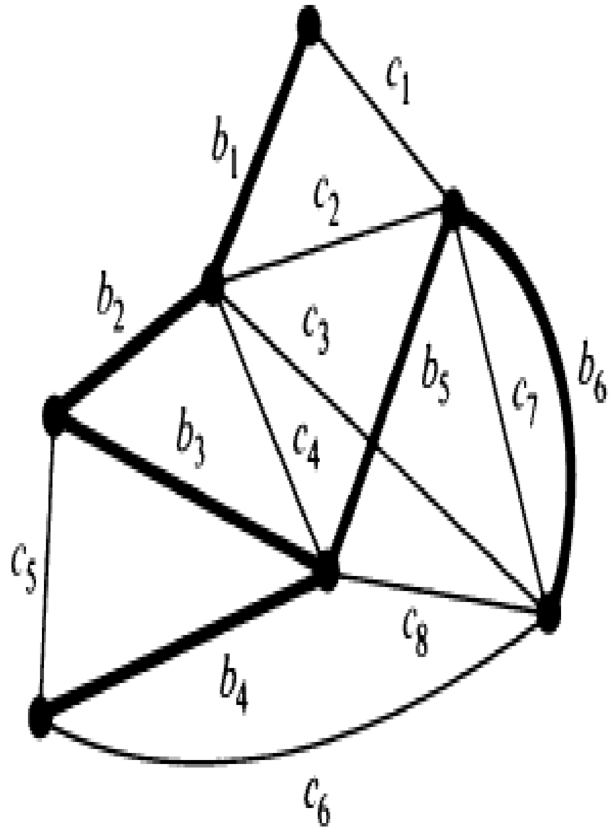
A connected graph G is a tree iff adding an edge between any two vertices in G creates exactly a circuit.

Proof: Consider a spanning tree T in G. Adding any one chord to T will exactly create one circuit.

Such a circuit formed by adding a chord in T is called ***fundamental circuit.***

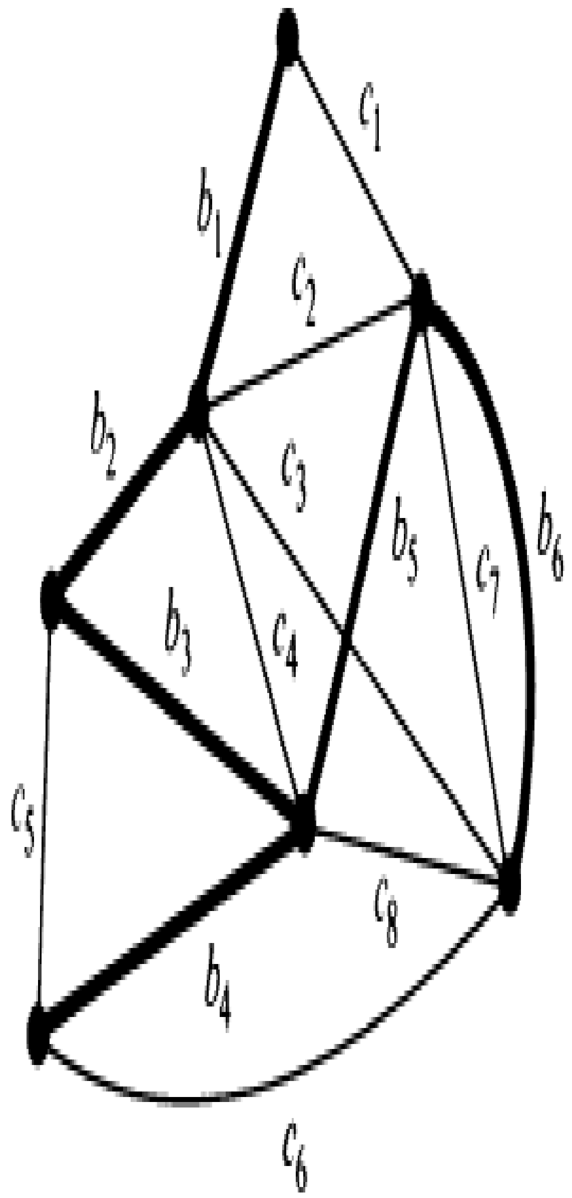
How many fundamental circuits in G????

e-n+k

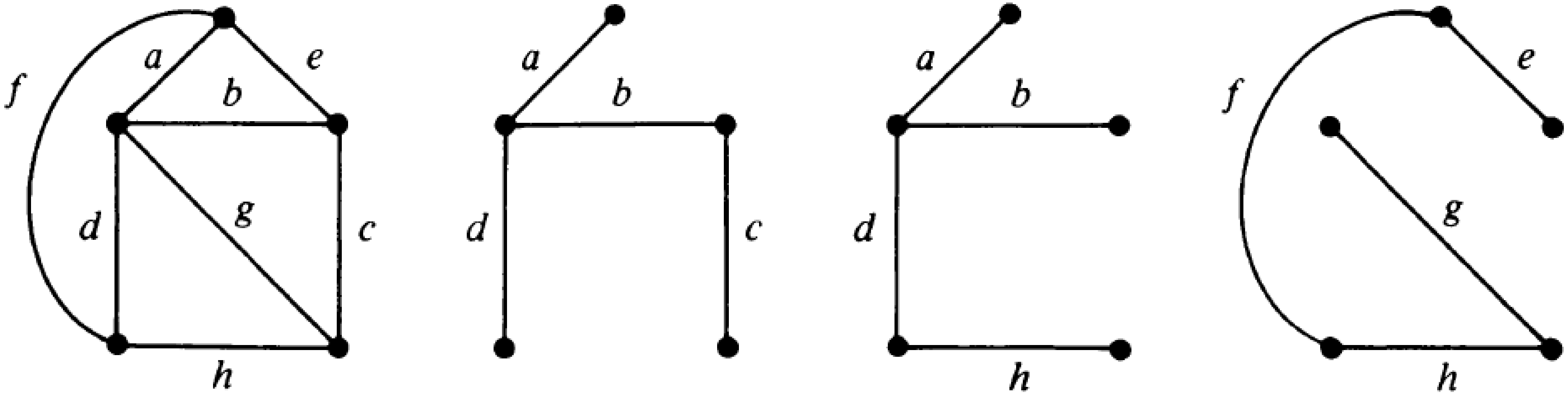


Circuits in a graph ???

- Ex: T={b1,b2,b3,b4,b5,b6}
- Add c1→ {b1,b2,b3,b4,b5,b6,c1}
- Fundamental Circuit→{b1,b2,b3,b5,c1}
- Add c2 to T → {b1,b2,b3,b4,b5,b6,c2}
- Fundamental Circuit→{b2,b3,b5,c2}
- Add c1 & c2 to T → {b1,b2,b3,b4,b5,b6,c1,c2}
- Circuits formed→ {b1,b2,b3,b5,c1} , {b2,b3,b5,c2}, {b1,c1,c2}



Finding all spanning trees of G



T_1 T_2 T_3

- The generation of one spanning tree from another , by adding a chord or deleting an appropriate branch is called ***cyclic interchange or elementary tree transformation.***

Distance between T_i & T_j

- Let T1 and T2 be spanning trees of the graph G.
- Defined as the number of edges of G present in one tree, but not in other.

$$\mathbf{d(T1, T2) = \tfrac{1}{2}[N(T1 \oplus T2)]}$$

Ring sum: Edges either in T1 or T2, but not in both

Theorem

The distance between the spanning trees of a graph is a metric.

$$d(T_i, T_j) \geq 0 \quad \text{and} \quad d(T_i, T_j) = 0 \text{ if and only if } T_i = T_j,$$

$$d(T_i, T_j) = d(T_j, T_i),$$

$$d(T_i, T_j) \leq d(T_i, T_k) + d(T_k, T_j).$$

Theorem

Starting from a spanning tree of a graph G, we can obtain every spanning tree of G by successive cyclic exchanges.

Proof: Since in a connected graph G of rank r, then a spanning tree T has r edges , following results:

1) ***max d(Ti,Tj)=1/2 max N[*(Ti ⊕ Tj)] ; …………………(eq 1)**

   <= r, the rank of G

2) **μ , nullity of G**

   no more than μ edges of a spanning tree can be replaced to get another spanning tree;

   max d(Ti, Tj)<= μ ……………………(eq 2)

- Thus by combining eq1 and eq 2 :

$$\max d(T_i, T_j) <= \min(\boldsymbol{\mu}, \mathbf{r})$$

Central tree

- For a spanning tree T_0 , of a graph G, let max $d(T_0 , T_i)$ denote maximal distance between T_0 and any other spanning tree of G. Then T_0 is called a central tree.

Tree Graph

- Defined as a graph in which each vertex corresponds to a spanning tree of G, and each edge corresponds to a cyclic interchange between spanning trees of G represented by the two end vertices of the edge.

Spanning trees in a weighted graph

- The weight of a spanning tree T is defined as the sum of the weights of all branches in T.
- Different spanning trees of G have different weights.
- A spanning tree with the smallest weight in a weighted graph is called ***a shortest spanning tree or shortest distance spanning tree or minimal spanning tree.***

Application

- To connect n cities v1, v2,….vn through a network of roads.
- The problem is to find the least expensive network that connects all n cities.

Theorem

A spanning tree T(of a given weighted connected graph G)is a shortest spanning tree (of G) iff there exists no other spanning tree (of G) at a distance of one from T whose weight is smaller than that of T.

Proof: Let X be any subset of the vertices of G, and let edge e be the smallest edge connecting X to G-X. Then e is part of the minimum spanning tree. Suppose we have a tree T not containing e; then we want to show that T is not the MST. Let e=(u,v), with u in X and v not in X. Then because T is a spanning tree it contains a unique path from u to v, which together with e forms a cycle in G. This path has to include another edge f connecting X to G-X. T+e-f is another spanning tree. It has smaller weight than T since e has smaller weight than f. So T was not minimum, which is what we wanted to prove.

Algorithms

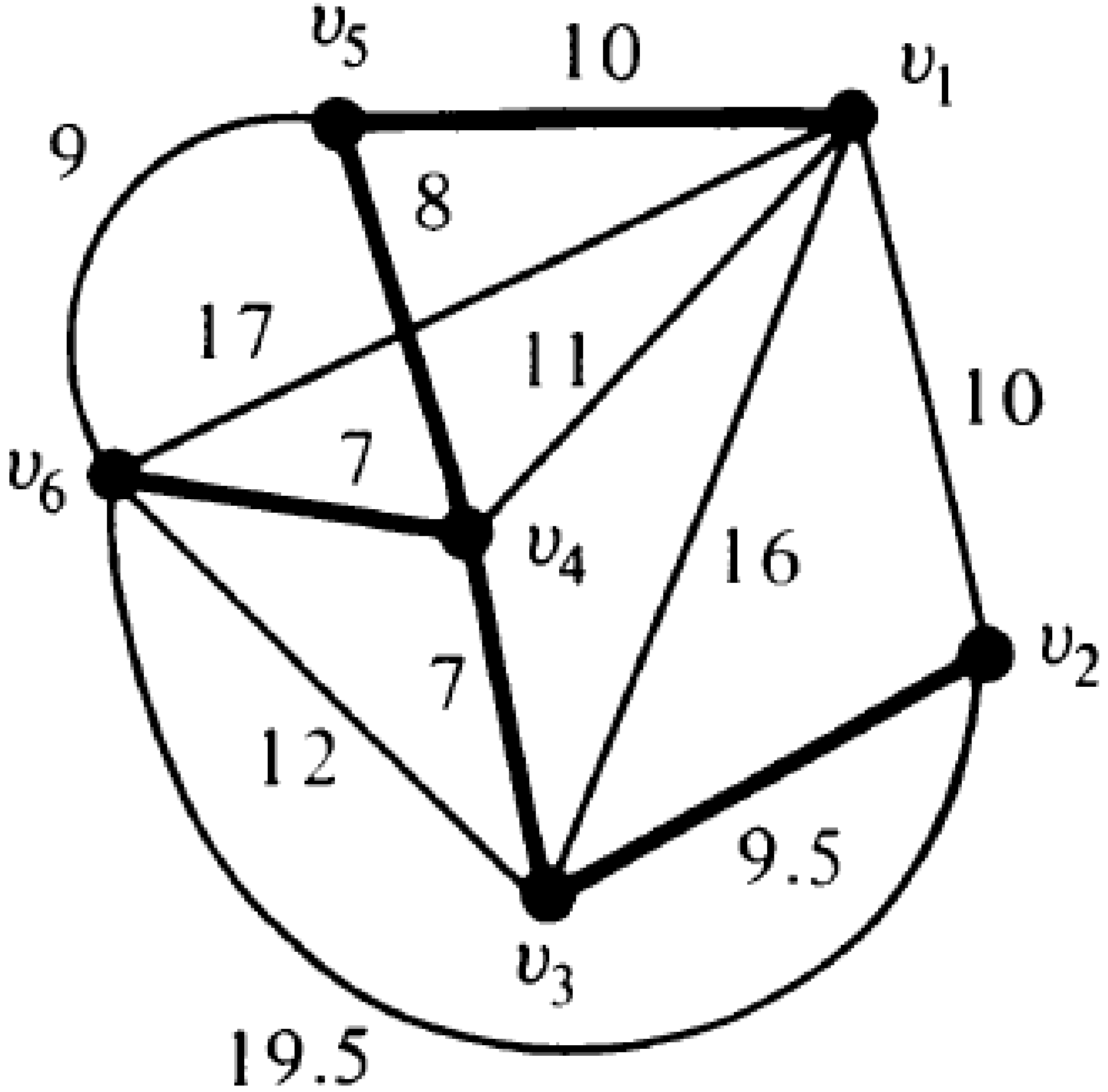
- **Kruskal's algorithm**
- **Prim's algorithm**

Kruskal's algorithm

- List all edges of G in increasing order of weights
- Select a smallest edge of G
- Then select for each successive step select another smallest weight edge which does not form any circuit with previously selected edges.
- Continue until n-1 edges are selected.
- This results in MST

Prim's algorithm

- It does not require listing of all edges in increasing order of weights
- And do not want to check whether newly adding edges create circuits
- Algorithm:
  - Draw n isolated vertices and label them as v1,v2,….vn
  - Tabulate the weights of each edges in an n by n table
  - Set weights of non existent edges as high
  - Start from v1 & connect it to nearest neighbor vk
  - v1- vk is now subgraph, connect this subgraph to its closest neighbour vi
  - Continue this process by connected n vertices using n-1 edges.


v_5
10
v_1
9
8
17
11
10
v_6
7
v_4
16
v_2
7
12
9.5
v_3
19.5

$$\begin{array}{cc}
 & \begin{array}{cccccc} \;v_1\; & \;v_2\; & \;v_3\; & \;v_4\; & \;v_5\; & \;v_6\; \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} &
\left[\begin{array}{cccccc}
- & 10 & 16 & 11 & 10 & 17 \\
10 & - & 9.5 & \infty & \infty & 19.5 \\
16 & 9\ 5 & - & 7 & \infty & 12 \\
11 & \infty & 7 & - & 8 & 7 \\
10 & \infty & \infty & 8 & - & 9 \\
17 & 19.5 & 12 & 7 & 9 & -
\end{array}\right]
\end{array}$$

Degree Constrained Spanning Tree

- In a MST, a vertex vi can have any degree 1<=d(vi)<= n-1.
- Consider we are giving a condition that each vertex cannot have more than degree three.

$$d(v_i) \le n-1 \text{ ; for every } v_i$$

- Such spanning tree is called Degree Constrained Spanning Tree.

- From a weighted graph G, find shortest spanning tree T such that,

  d(vi)<=2.