DAY 9

Fundamental Circuit

Let us assume a spanning tree T in a graph G. Adding any **one** chord to T will create exactly one circuit and that circuit is known as a fundamental circuit.

How many fundamental circuits does a graph have??

Exactly as many as the number of chords, i.e. , (e-n+1)

A circuit is fundamental only with respect to a spanning tree.

Fundamental Circuit

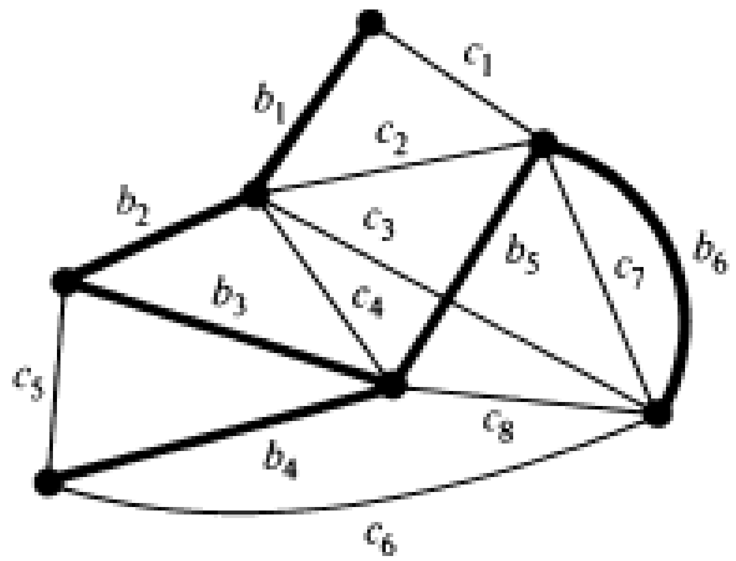


Fig. 3-17 Spanning tree.

Elementary Tree Transformation

Usually, in a given connected graph there are a large number of spanning trees. In many applications we require all spanning trees. One reasonable way to generate spanning trees of a graph is to start with a given spanning tree, say tree T_1 ($a\ b\ c\ d$ in Fig. 3-19). Add a chord, say h, to the tree T_1. This forms a fundamental circuit ($b\ c\ h\ d$ in Fig. 3-19). Removal of any branch, say c, from the fundamental circuit $b\ c\ h\ d$ just formed will create a new

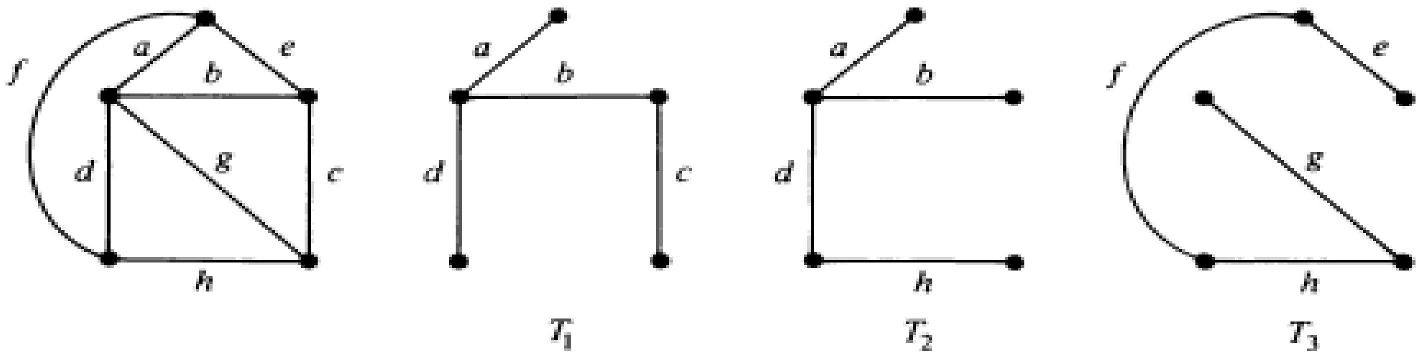
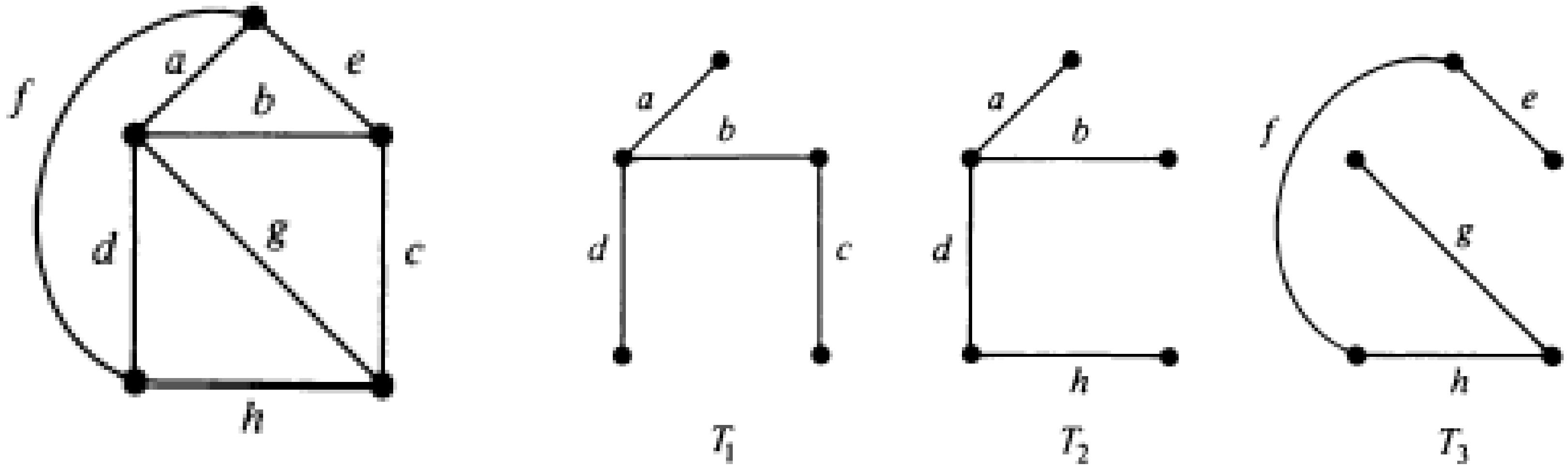


Fig. 3-19 Graph and three of its spanning trees.

spanning tree T_2. This generation of one spanning tree from another, through addition of a chord and deletion of an appropriate branch, is called a *cyclic interchange* or *elementary tree transformation*. (Such a transformation is a standard operation in the iteration sequence for solving certain transportation problems.)

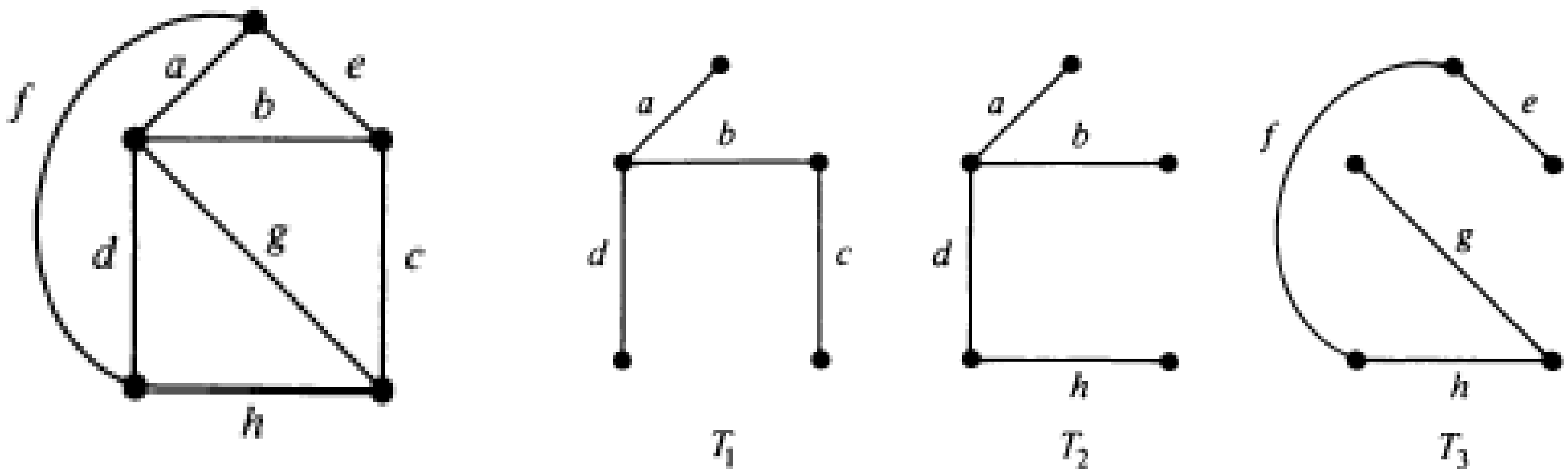
Distance between spanning trees



The distance between two spanning tree is defined as the number of edges present in one tree but not in the other.

d(T1, T2) = 1,
d(T1, T3) = 4
d(T3, T2) = 3

Solve



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

Minimum Spanning Tree (MST)

A spanning Tree with the smallest weight in a weighted graph is called a shortest spanning tree or minimum spanning tree.

Algorithms for finding MST

Kruskal's Algorithm

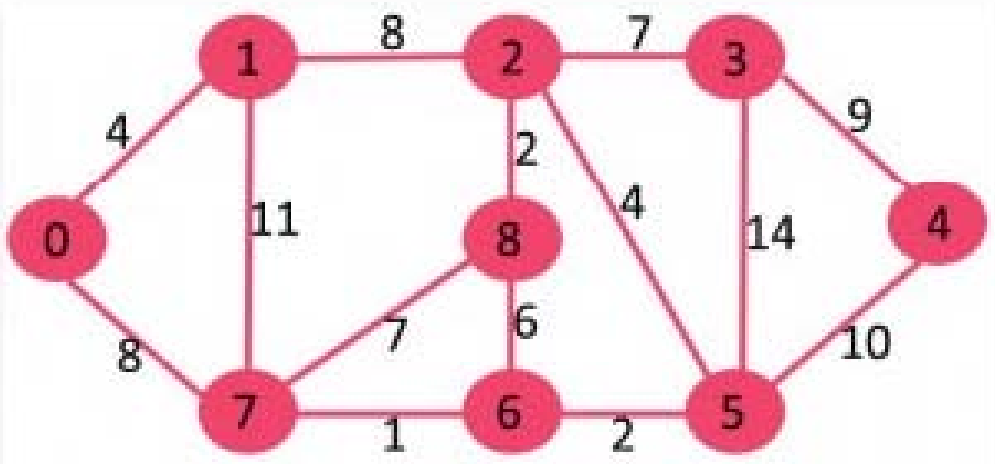
Prim's Algorithm

Kruskal's Algorithm

The steps for finding MST using Kruskal's algorithm:

1. *Sort all the edges in non-decreasing order of their weight.*
2. *Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.*
3. *Repeat step 2 until there are (n-1) edges in the spanning tree.*

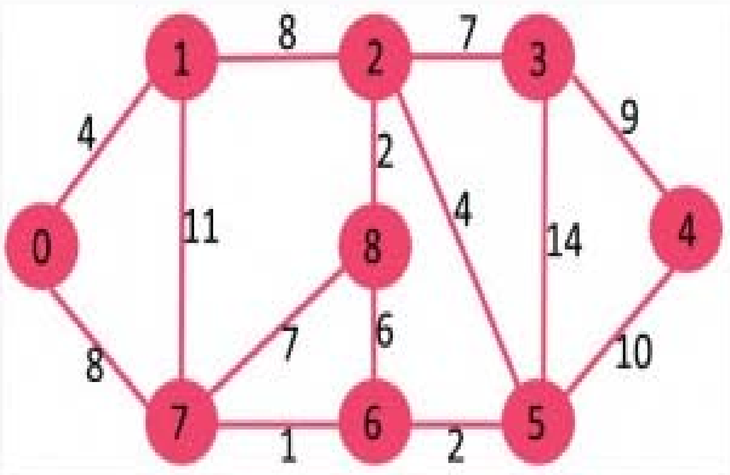
Kruskal’s Algorithm



The graph contains 9 vertices and 14 edges. So, the minimum spanning tree formed will be having (9 – 1) = 8 edges.

Contd..

1. *Sort all the edges in non-decreasing order of their weight.*



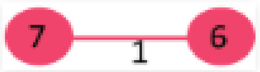

```
After sorting:
Weight   Src    Dest
1         7      6
2         8      2
2         6      5
4         0      1
4         2      5
6         8      6
7         2      3
7         7      8
8         0      7
8         1      2
9         3      4
10        5      4
11        1      7
14        3      5
```

Contd..

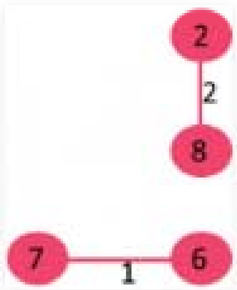
2. *Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.*

Now pick all edges one by one from sorted list of edges

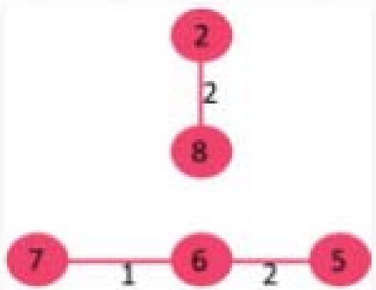
1. *Pick edge 7-6:* No cycle is formed, include it.

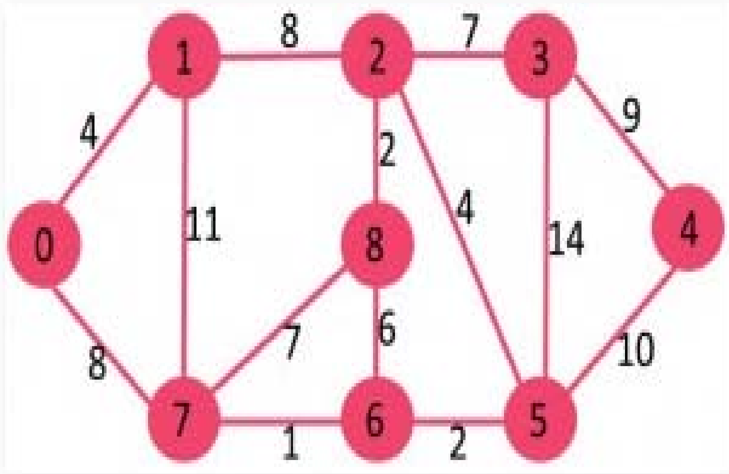


2. *Pick edge 8-2:* No cycle is formed, include it.



3. *Pick edge 6-5:* No cycle is formed, include it.

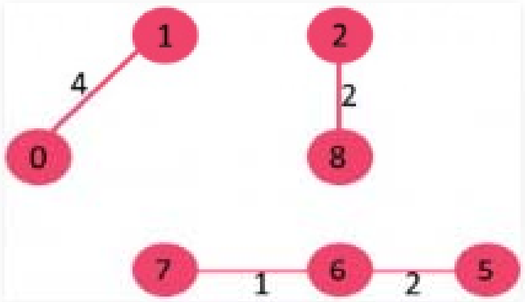


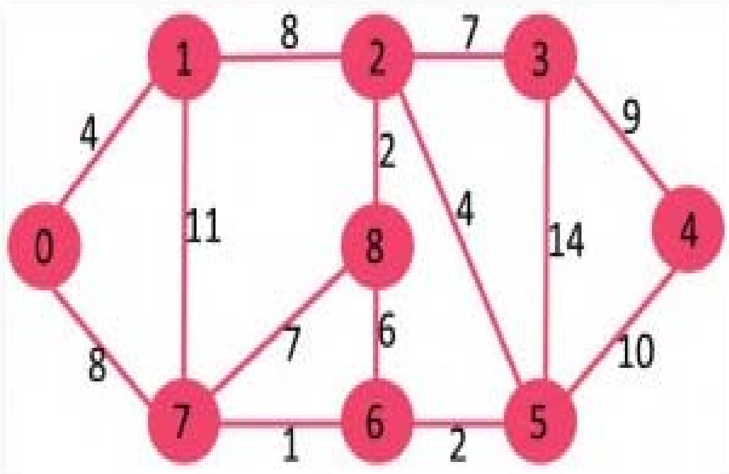


Contd..

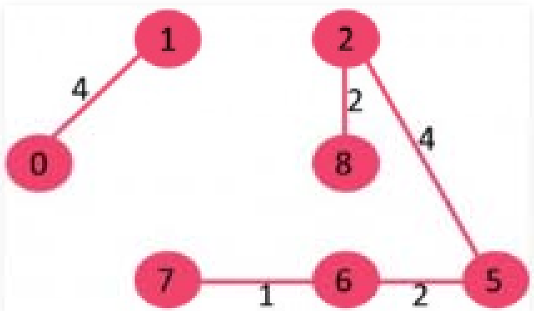
***2.** Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.*

4. *Pick edge 0-1:* No cycle is formed, include it.





5. *Pick edge 2-5:* No cycle is formed, include it.

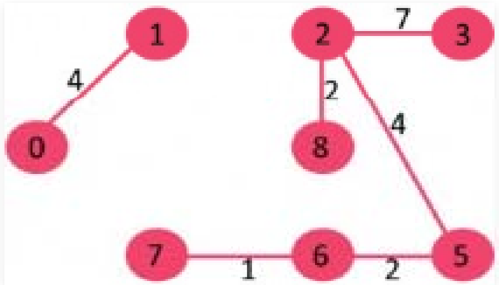


6. *Pick edge 8-6:* Since including this edge results in cycle, discard it.

Contd..

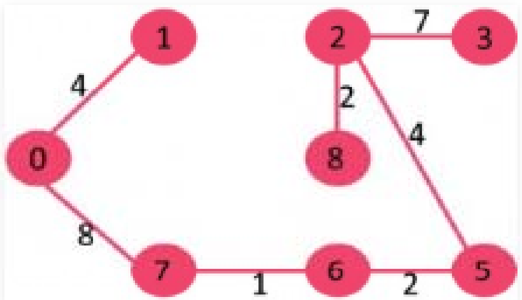
***2.** Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.*

7. *Pick edge 2-3:* No cycle is formed, include it.

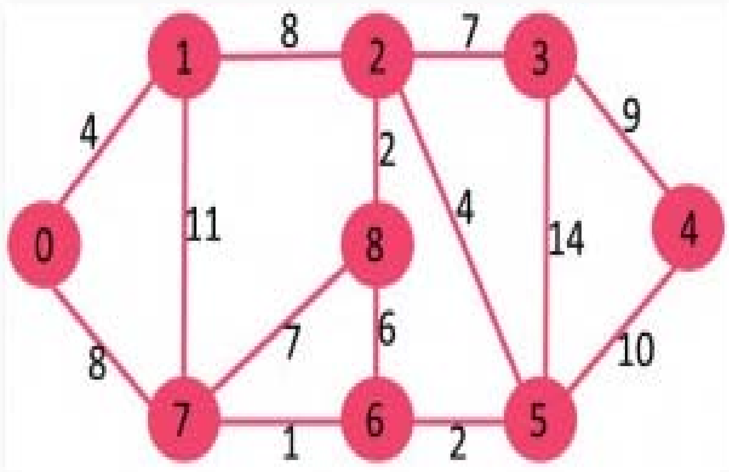


8. *Pick edge 7-8:* Since including this edge results in cycle, discard it.

9. *Pick edge 0-7:* No cycle is formed, include it.



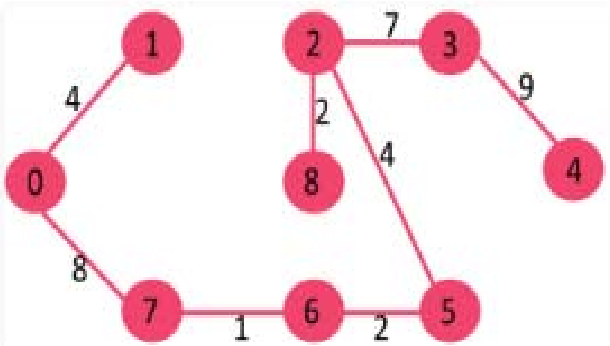
10. *Pick edge 1-2:* Since including this edge results in cycle, discard it.



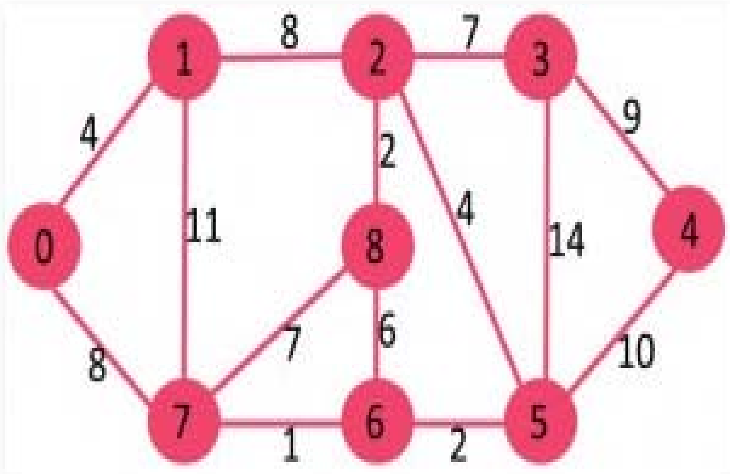
Contd..

3. *Repeat step 2 until there are (n-1) edges in the spanning tree.*

11. *Pick edge 3-4:* No cycle is formed, include it.



Since the number of edges included equals (V – 1), the algorithm stops here.

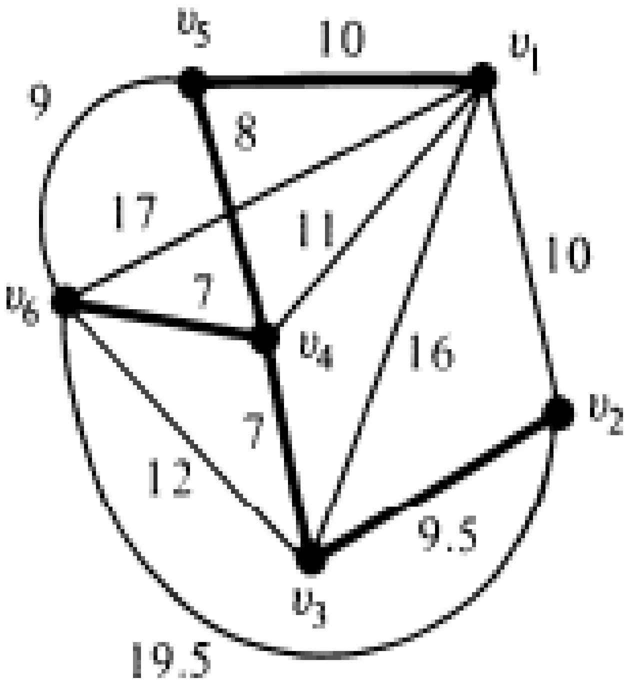


Prim's Algorithm

For Prim's algorithm, draw n isolated vertices and label them $v_1, v_2, \ldots, v_n$. Tabulate the given weights of the edges of G in an n by n table. (Note that the entries in the table are symmetric with respect to the diagonal, and the diagonal is empty.) Set the weights of nonexistent edges (corresponding to those pairs of cities between which no direct road can be built) as very large.

Start from vertex v_1 and connect it to its nearest neighbor (i.e., to the vertex which has the smallest entry in row 1 of the table), say v_k. Now consider v_1 and v_k as one subgraph, and connect this subgraph to its closest neighbor (i.e., to a vertex other than v_1 and v_k that has the smallest entry among all entries in rows 1 and k). Let this new vertex be v_i. Next regard the tree with vertices v_1, v_k, and v_i as one subgraph, and continue the process until all n vertices have been connected by $n - 1$ edges. Let us now illustrate this method of finding a shortest spanning tree.

Contd..

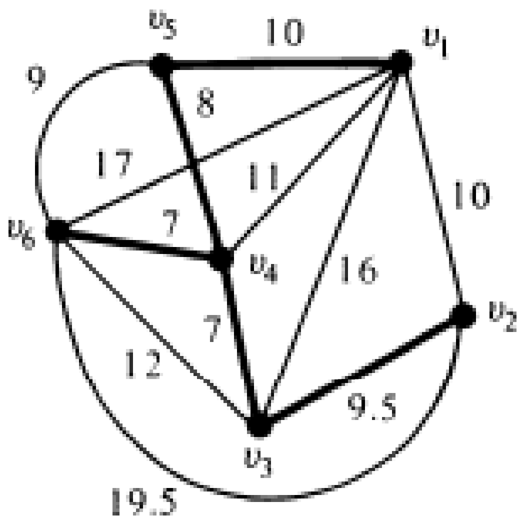


(a)

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

(b)

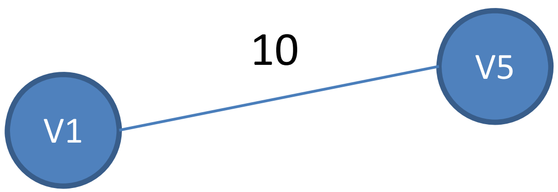
Contd..



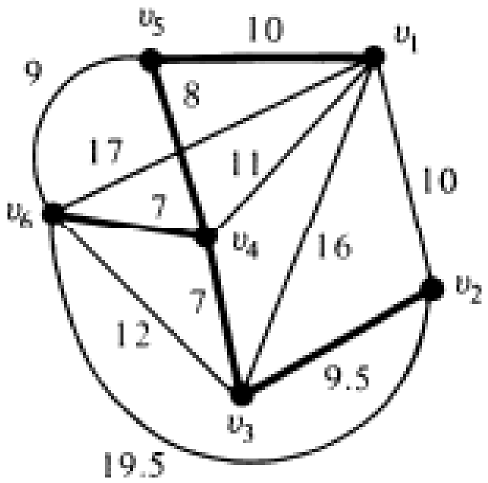
(a)

	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	–

(b)



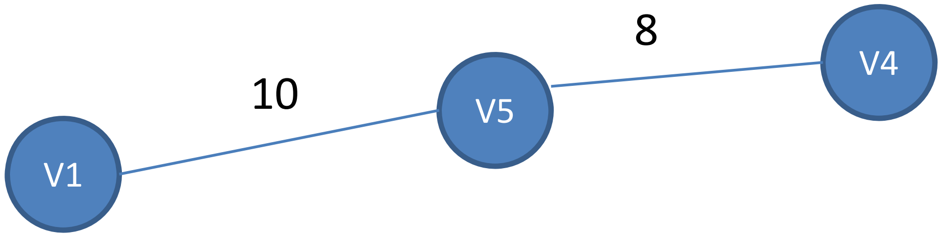
Contd..



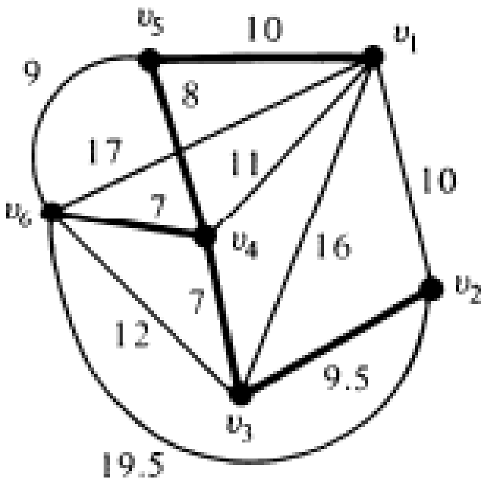
(a)

$$
\begin{array}{c|cccccc}
 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\
\hline
v_1 & - & 10 & 16 & 11 & 10 & 17 \\
v_2 & 10 & - & 9.5 & \infty & \infty & 19.5 \\
v_3 & 16 & 9.5 & - & 7 & \infty & 12 \\
v_4 & 11 & \infty & 7 & - & 8 & 7 \\
v_5 & 10 & \infty & \infty & 8 & - & 9 \\
v_6 & 17 & 19.5 & 12 & 7 & 9 & -
\end{array}
$$

(b)



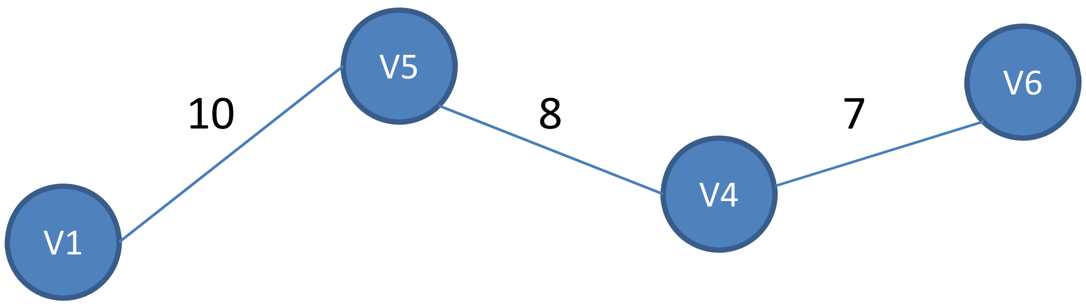
Contd..



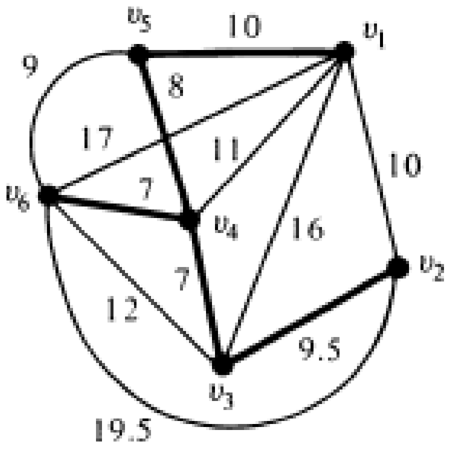
(a)

$$
\begin{array}{c|c}
 & \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} &
\begin{bmatrix}
- & 10 & 16 & 11 & 10 & 17 \\
10 & - & 9.5 & \infty & \infty & 19.5 \\
16 & 9.5 & - & 7 & \infty & 12 \\
11 & \infty & 7 & - & 8 & \boxed{7} \\
10 & \infty & \infty & 8 & - & 9 \\
17 & 19.5 & 12 & \boxed{7} & 9 & -
\end{bmatrix}
\end{array}
$$

(b)



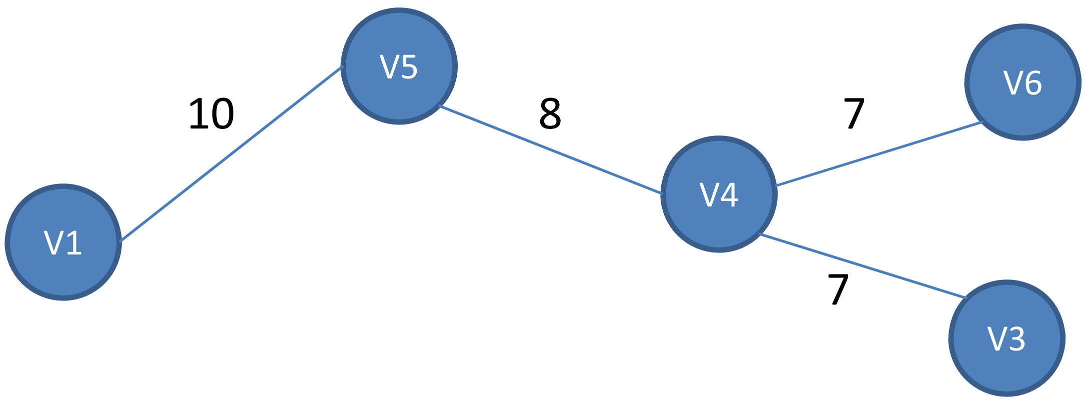
Contd..



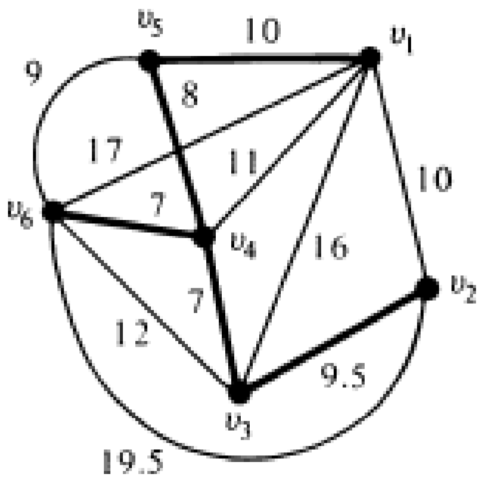
(a)

$$
\begin{array}{c|cccccc}
 & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\
\hline
v_1 & - & 10 & 16 & 11 & 10 & 17 \\
v_2 & 10 & - & 9.5 & \infty & \infty & 19.5 \\
v_3 & 16 & 9.5 & - & 7 & \infty & 12 \\
v_4 & 11 & \infty & 7 & - & 8 & 7 \\
v_5 & 10 & \infty & \infty & 8 & - & 9 \\
v_6 & 17 & 19.5 & 12 & 7 & 9 & -
\end{array}
$$

(b)



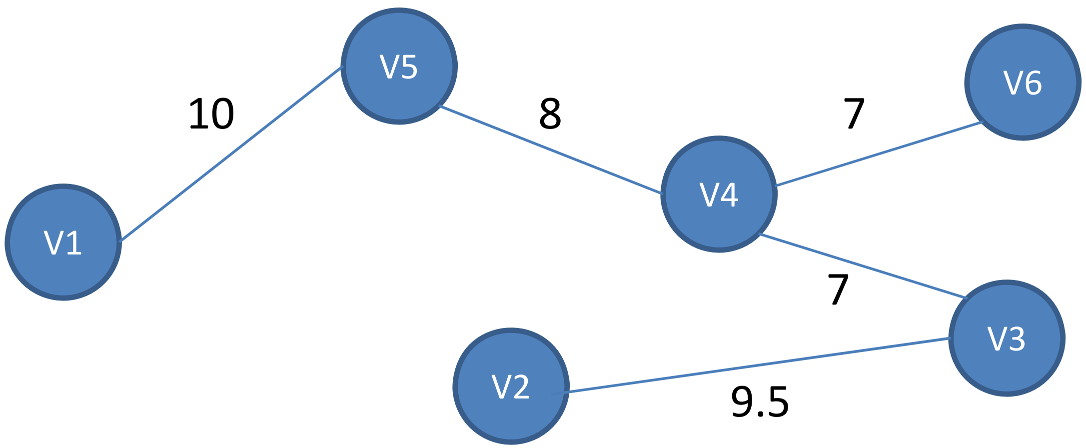
Contd..



(a)

	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	–

(b)



Total Weight
=10+8+7+7+9.5
= 41.5