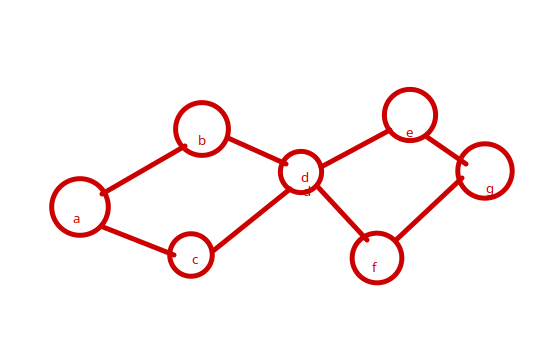
Algorithm for constructing spanning tree

BFS[breadth first searched] algorithm



I)Chose a vertex to be a root

Ii) add edges from vertex ‘a’ to all the adjacent vertex on it

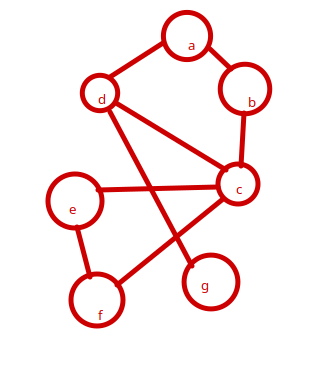
Iii) add edges from this vertices from level 1 to adjacent vertices which are not already joint in the tree.

Iv) add edges from d in level 2 to adjacent vertices not already in tree.

V) add edges from e at level 3 to adjacent vertices which are not already in tree.

Vi) stop if all the vertices are connected.

DFS[depth search search(back tracking)]



Pseudo code:

DFS(G,S)

{

T={s}

Traverse (s)

}

Traverse (v)

{

For each w adjacent to v and not yet in tree

{

T V{w} //put edge (v,w) also

Traverse (w)

}

}

}

Minimal spanning tree:

The weight of a spanning tree of graph G is a spanning tree with minimal possible weight among all all possible spanning trees of that graph.

There are many algorithms to trace the minimal spanning trees among them we will discuss two.

1.Kruskal’s Algorithm

Pseudo code:

Kruscal\_mst(G)

{

T = {v} //set of vertices

E = set of edges sorted in non decreasing order of there weight while |T| < n-1 and E!=Null

{

Select (u,v) from E in order

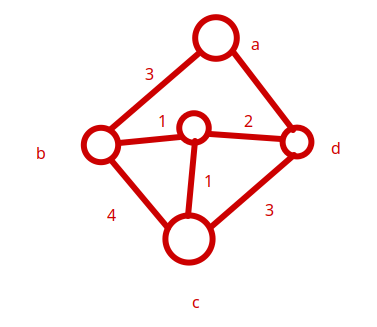
Remove (u,v) from E

If (u,v) doesnot create cycle in T

T = T U (u,v)

}

}



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| B,C | C,B | C,D | B,A | D,E | A,D | B,E |
| 1 | 1 | 2 | 3 | 3 | 4 | 4 |

