International Journal of Mathematical Archive-9(8), 2018, 40-43
Available online through www.ijma.info ISSN 2229 – 5046

MINIMUM SPANNING WITH DIJKSTRA'S ALGORITHM

J. MOHAN[1], S. PRABHAVATHI[2], R. MUTHUKUMAR[3], N. SANGEETHA[4], C. AROCKIYADASS[5] AND S. DICKSON*[6]

[1,2,3,4,5,6]**Assistant Professor of Mathematics, Vivekanandha Educational Institutions, India.**

(Received On: 02-06-18; Revised & Accepted On: 01-08-18)

ABSTRACT

A *tree is an undirected graph in which any two vertices are connected by exactly one path. There are many kinds of trees, but in this paper to discussing about what is a minimum spanning tree and how it can be applied in real life situations.*

Key Words: *Connected graph, undirected graph, spanning tree, sub-graph, edges, vertex.*

1. INTRODUCTION

Graph is a platform for many structures. For example, we can create a shortest route from one city to another city by using map. In this article, we are discussing how to find the shortest route. One way of finding the shortest route is by Dijstra's Algorithm. Let us discuss the algorithm before that minimum spanning tree have direct applications in the design of networks, including computer networks, telecommunications networks, transformation networks water supply networks and electrical grids

2. BASIC CONCEPTS

2.1 Graph and Simple Graph:

A Simple graph G is an ordered pair a= (V, E) = (v(G), E(G)), where V is a non-empty set containing information regarding the vertices of G and E is a set E $\subseteq$ [V] ^2 containing the information on the edges in G.(1)

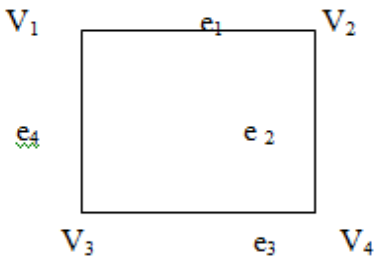


Figure-2.1

2.2 Sub-graphs

A Sub-graph S of a graph G is a graph whose vertex set V(S) is a subset of the vertex set V(G), ie., V(S) $\subseteq$ V(G), and whose edge set E(S) is a subset of the edge set E(G).

i.e., E(S) $\subseteq$ E(G).

Corresponding Author: S. Dickson*[6]
[6]Assistant Professor of Mathematics, Vivekanandha Educational Institutions, India.

Example, the following graph S is a sub-graph of G

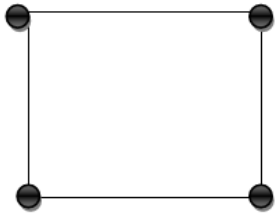


Figure-2.2

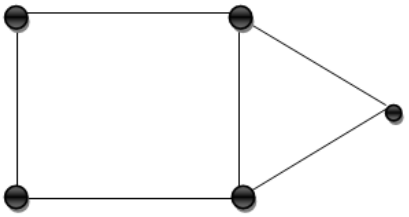


Figure-2.3: Sub-graph of G

2.3 Undirected graph:

Undirected graphs have edges that do have a direction, the edges indicates a two-way relationship, in that each edge can be travelled in both directions. Maximum number in an undirected graph without a loop is n(n-1)/2.

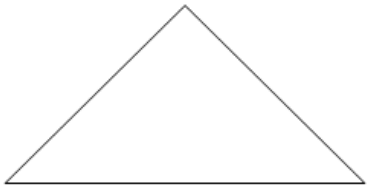


Figure-2.4: Undirected graph

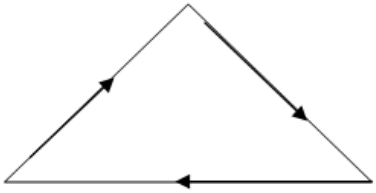


Figure-2.5: Directed graph

2.4 Spanning Tree:
Spanning Tree of a graph is the minimal connected sub-graph of the graph which contains all the vertices of the given graph with minimum possible number of edges.

Spanning Tree does not contain cycles and it also be connected.

A graph can have more than one spanning tree, suppose t6he given graph is having n vertices then its spanning tree consists of n-1 edges and n vertices.

Spanning Tree is minimally connected. It infers than on removing any edge from spanning tree, the graph will become disconnected.

2.5 Minimum Spanning Tree:
Minimum Spanning Tree is a spanning tree that has minimum weight than all other spanning trees of the same graph. Weight of a spanning tree is calculated by adding all the weights of the edges of the spanning tree. A given graph has a unique minimum spanning tree. So a given graph can have more than one spanning trees but has a unique minimum spanning tree.

Two most important algorithm for calculating minimum spanning tree are: Kruskal's Algorithm and Prim's Algorithm.

2.6 Dijkstra's Algorithm:
It is an algorithm for finding the shortest paths between nodes in a graph, which many represent, road networks. Dijkstra's algorithm is very similar to Prim's algorithm for minimum spanning tree.

3. ALGORITHM

1) Create a set shortest path Tree (SPT) set that keeps track to vertices included in shortest path tree, i.e., whose minimum distance from source is calculated and finalized. Initially, this set is empty.
2) Assign a distance value to all vertices in the input graph. Initialize all distance source vertex as infinite. so that it is picked first
3) While shortest path tree set does not include all vertices.
   a) Pick a vertex U which is not there in SPT set and has minimum distance value.
   b) Include U to SPT set.
   c) Update distance value of all adjacent vertices of U. To update the distance values , iterate through all adjacent vertices for every adjacent vertex V, if sum of distance value of U and weight of edge U-V, is less than the distance value of V, then update the distance value of V.

Example:

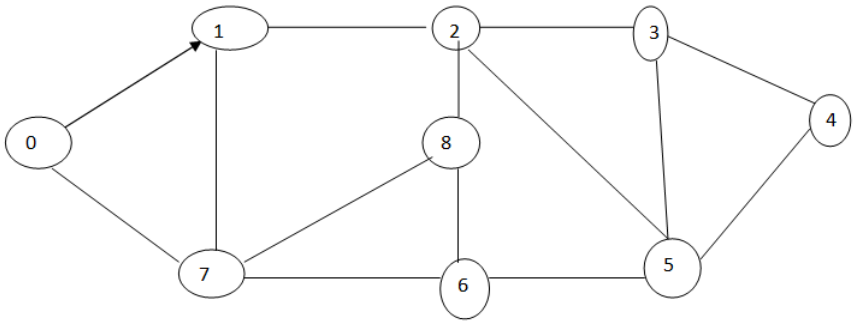


Figure-2.6

The set SPT set is initially empty and distances assigned to vertices are {0, INF, INF, INF, INF, INF, INF} where INF indicate infinite.

Now pick the vertex with minimum distance value. The vertex 0 is picked, include it in SPT set .So SPT set becomes {0} .After including 0 to SPT set, update distance values of its adjacent vertices.
Adjacent vertices of 0 are 1 and 7.

The distance values of 1 and 7 are updated as 4 and 8

Following sub-graph shows vertices and distance values, only the vertices with finite distance values are shown .The vertices included in SPT are shown is green colour.

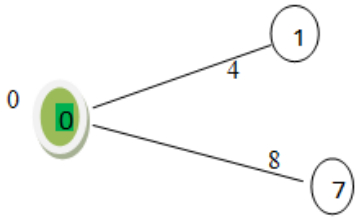


Figure-2.8

Pick the vertex with minimum distance value and not already included in SPT set. So SPT set now becomes {0, 1}. Update the distance values of adjacent vertices of 1. The distance value of vertex 2 becomes 2

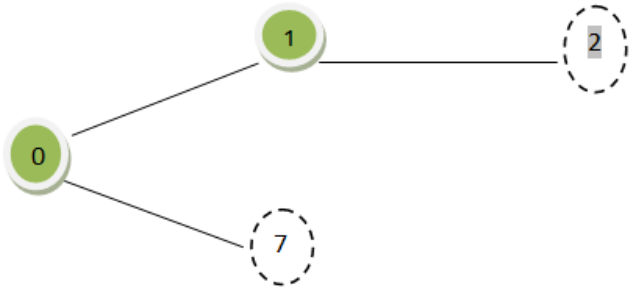


Figure-2.9

Pick the vertex with minimum distance value and not already included in SPT. Vertex 7 is picked. So SPT set now becomes {0, 1, 7}. Update the distance values of vertex 6 and 8 becomes finite (15 and 9 respectively)

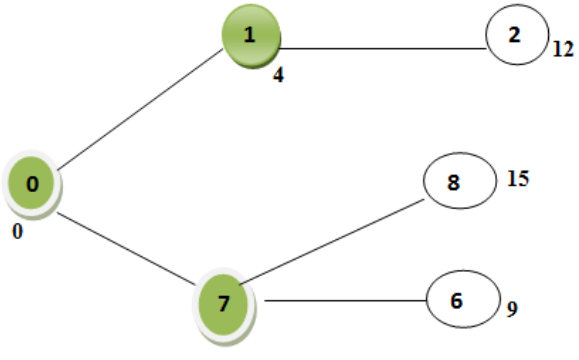
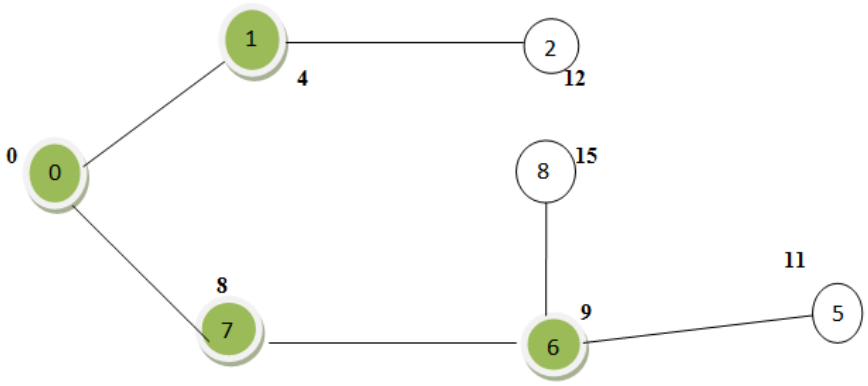
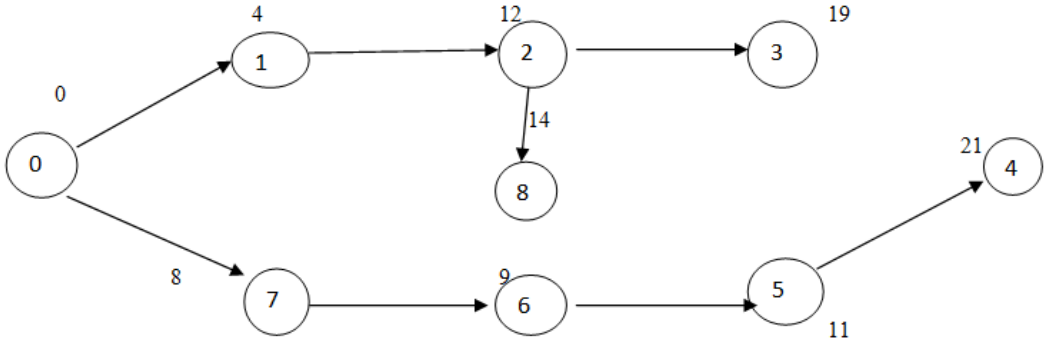


Figure-2.10

Pick the vertex with minimum distance value and not already included in SPT. Vertex 6 is picked. So SPT set now becomes {0, 1, 7, 6}. Update the distance values of adjacent vertices of 6.The distance value of vertex 5 and 8 are updated.



We repeat the above steps until SPT set doesn't include all vertices of given graph. Finally, we get the following shortest path tree (SPT).



In Dijkstra's algorithm, the distance from one node to another is the sum of all intermediate edges from root to destination.

CONCLUSION

IF we have to add 1+2+3+4+5+6+7, if takes little bit of time, but what if we say add 10 to the result it will not take time because you already have stored the previous result in your mind. This is the concept that used by Dynamic programming to make recursive program a lot more efficient.

REFERENCE

1. http://www.Mathonline.wikidot.com
2. www.teetorials.com
3. www.mathworks.com
4. http://www.en.m.wikipedia.org
5. www.quora.com

Source of support: Nil, Conflict of interest: None Declared.