International Journal of Multidisciplinary and Current Research ISSN: 2321-3124

Research Article Available at: http://ijmcr.com

Generation of Minimal Spanning Tree using Weight Matrix and concept of Cut Edge

Babita Bist[*]

Assistant Professor, Department. of Mathematics, S. D. College (Lahore), Ambala Cantt, India

Received 18 July 2020, Accepted 15 Aug 2020, Available online 22 Aug 2020, **Vol.8 (July/Aug 2020 issue)**

Abstract

Minimal Spanning tree (MST) is a fundamental problem in networking design. The problem of constructing MST of an undirected connected weighted graph belongs to the group of classical combinatorial optimization problems. This paper presents minimum cost (weight) spanning tree using the algorithm which is based on the weight matrix of weighted graph and the concept of cut edge. The algorithm particularly uses only upper triangular part of weight matrix.

Keywords: *Graph, Weight Graph, Spanning tree, Minimum Cost Spanning Tree, Cut Edge*

Introduction

A graph is a collection of vertices and edges, and each edge connects a pair of vertices. A tree is an acyclic connected graph. A spanning tree of a graph is a tree that is a sub-graph of the graph and contains all the vertices of the graph. When the edges of a graph are associated with some weights which may represent distance, cost or time etc., it is called a weighted graph. A minimum spanning tree of a weighted graph is a tree of the graph which contains all the vertices of the graph and the sum of weights of all its edges is minimum among all such possible trees of the graph. MST Tree must be finding from the Graph. There can be multiple MST of a graph, but all of these MSTs must have unique same total cost. The minimum spanning tree problem is to construct a minimum cost spanning tree with the weight graph and is significant to network optimization. A common problem in communication networks and circuit design is that of connecting together a set of nodes by a network of minimal total length. To minimize the length of connecting network, it never pays to have cycle, since we could break any cycle without breaking the connectivity.

Application of MST

Minimum Spanning Tree is fundamental problem with wide applications in different fields. It represents many complicated real-world problems. The standard application of MST is in network design i.e. in telephone networks, TV cable network, computer network, road network, islands connection, pipeline network, electrical circuits, utility circuit printing, obtaining an independent set of circuit equations for an electrical network etc.

If you have a business with several offices; you want to lease phone lines to connect them up with each other; and the phone company charges different amounts of money to connect different pairs of cities. You want a set of lines that connects all your offices with a minimum total cost. It should be a spanning tree, since if a network isn't a tree you can always remove some edges and save money. MST also offers a method of solution to problems such as network reliability, clustering and classification problems and also used to find the approximation solution for the NP hard problems. A less obvious application is that the minimum spanning tree can be used to approximately solve the traveling salesman problem.

Objective of MST

- To minimize cost of the spanning tree for both directed and undirected.
- To minimize load on the network.
- To eliminate the cycle from the graph from the MST.
- To improve the complexity of the MST

Boruvka's algorithm

Borůvka's algorithm is a method for finding a minimum spanning tree for which all edge weights are distinct. It is oldest minimum spanning tree algorithm discovered long before computers. The algorithm was published as a method of constructing an efficient electricity network. The algorithm was first invented by Otakar Boruvka in 1926 when he was trying to find an optimal routing for the electrical grid in Moravia. Later re-discovered by Choquet in 1938, then by Florek, Perkal and Zubrzycki in 1951 and again discovered by Sollin in 1965. The algorithm is frequently called Sollin's algorithm.

*Corresponding author's ORCID ID: 0000-0000-0000-0000
DOI: https://doi.org/10.14741/ijmcr/v.8.4.10

The algorithm is based on merging of disjoint components. At the beginning each vertex is considered as a separate component. In each step the algorithm connects every vertex to some other by using the cheapest edge. It will continue joining these edges until all vertices is visited in spanning tree. The algorithm taken O (log V) iterations of the outer loop until it terminates. Therefore, it take O (E log V) time to run. Where E is the number of edges, and V is the number of vertices in graph. A significant advantage of this algorithm is that it can be easily parallelized because the choice of the cheapest path for each vertex is independent of the choice made by other vertex.

Prim's algorithm

The algorithm was first developed in 1930 by Czech mathematician Vojtěch Jarník and later rediscovered by Robert C, then by Prim in 1957 and Edsger W. Dijkstra in 1959. Therefore, it is also called the **DJP algorithm**, the **Prim–Jarník algorithm**, or the **Prim–Dijkstra algorithm**. Prim's algorithm starts with an empty spanning tree. Starting node is selected randomly from given graph. A set which contain all the edges in the graph is created. The edge, which is adjacent to the randomly selected node and with minimum weight among all adjacent edge, is selected from the set and added to new graph if it connects a vertex in the tree with a vertex not in the tree. This process is repeated until every edge in the set connects two vertices in the tree. Time complexity of this algorithm is O (E lg V). Prim's algorithm is significantly faster in the limit when got a really dense graph with many more edges than vertices and also easy to understand.

Modified Prim's algorithm

This algorithm is modified version of the prim's algorithm. In Prim's algorithm chooses a root node randomly and starts processing but in modified prim's algorithm minimum weighted edge is selected first. This algorithm is slightly different than prim's algorithm. In modified prim's algorithm select minimum weighted edge so it gives slightly better performance than original prim's algorithm. Time complexity of this algorithm is same as prim's algorithm.

Kruskal algorithm

This minimum spanning tree algorithm was first described by Kruskal in 1956 in the same paper where he rediscovered Jarnik's algorithm. The algorithm first appeared in Proceedings of the American Mathematical Society during 1956. The basic idea of the Kruskal's algorithms is to scan all the edges in increasing weight order; if an edge is safe, keep it. It is a minimum spanning tree algorithm which finds an edge of the least possible weight that connects any two trees in the forest.

In Kruskal's algorithm all edges are shorted in increasing order and selected the lowest edges first for becoming a minimum spanning tree. If there is a cycle generated then selected edges will be removed from the graph and next lowest edges are selected. The process will repeat till (n-1) edges will be added in to the graph. Using simple data structure Kruskal's algorithm complexity is O (E log V) time. Where E is the number of edges in the graph and V is the number of vertices. A significant advantage of this algorithm is that it can give good result for large number of vertices and edges also. But its complexity increases in case of same weights.

Construction of Minimum Cost Spanning Tree using Matrix

Let G(V, E, W) be a connected weighted graph with n vertices, where V is the set of vertices, E is the set of edges and W be the set of weights to the respective edges of the graph.

Let e_{ij} be the edge adjacent to vertices v_i and v_j and w_{ij} be the weight associated to the edge e_{ij}.

The Weight Matrix M of the graph G is defined as follows:

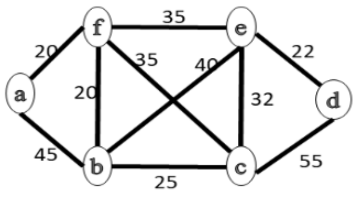
$$M_{ij} = \begin{cases} w_{ij} \ ; & \text{If there is an edge between the vertices } v_i \text{ to } v_j \text{ in G} \\ 0 \ ; & \text{otherwise} \end{cases}$$

Algorithm

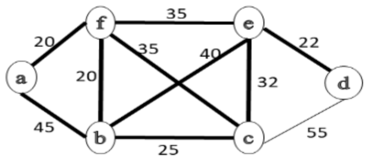
1. Prepare weight matrix according to definition given above.
2. Search the weight matrix M either column-wise or row-wise to find the unmarked nonzero maximum element M_{ij} which is the weight of the corresponding edge e_{ij} in M.
3. If the selected edge e_{ij} of M_{ij} is cut edge then Mark M_{ij} Else Set M_{ij} = 0
4. Repeat Step 2 to Step 3 until all n(n-1)/2 elements matrix of M are either marked or set to zero. Construct the spanning Tree T having only the marked elements from the weight matrix M which shall be the Minimum cost spanning tree of G.

Example

Let we consider the weighted graph with six vertices and ten edges. The weight matrix is as follows:

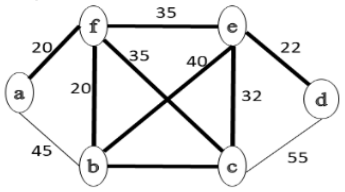


$$\begin{bmatrix} 0 & 45 & 0 & 0 & 0 & 20 \\ 45 & 0 & 25 & 0 & 40 & 20 \\ 0 & 25 & 0 & 55 & 32 & 35 \\ 0 & 0 & 55 & 0 & 22 & 0 \\ 0 & 40 & 32 & 22 & 0 & 35 \\ 20 & 20 & 35 & 0 & 35 & 0 \end{bmatrix}$$



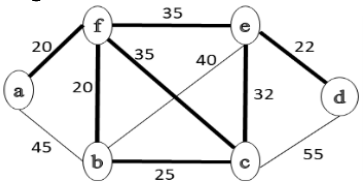
$$\begin{bmatrix} 0 & 45 & 0 & 0 & 0 & 20 \\ 45 & 0 & 25 & 0 & 40 & 20 \\ 0 & 25 & 0 & \textcircled{0} & 32 & 35 \\ 0 & 0 & 55 & 0 & 22 & 0 \\ 0 & 40 & 32 & 22 & 0 & 35 \\ 20 & 20 & 35 & 0 & 35 & 0 \end{bmatrix}$$

The largest element 55 is marked as zero and the corresponding edge (c, d) is removed as it is not a cut edge.



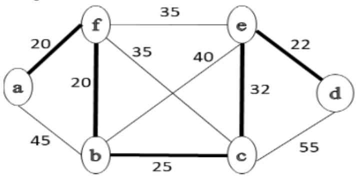
$$\begin{bmatrix} 0 & \textcircled{0} & 0 & 0 & 0 & 20 \\ 45 & 0 & 25 & 0 & 40 & 20 \\ 0 & 25 & 0 & 0 & 32 & 35 \\ 0 & 0 & 55 & 0 & 22 & 0 \\ 0 & 40 & 32 & 22 & 0 & 35 \\ 20 & 20 & 35 & 0 & 35 & 0 \end{bmatrix}$$

The next largest element 45 is marked as zero and the corresponding edge (a, b) is removed as it is not a cut edge.



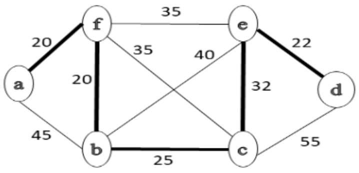
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 20 \\ 45 & 0 & 25 & 0 & \textcircled{0} & 20 \\ 0 & 25 & 0 & 0 & 32 & 35 \\ 0 & 0 & 55 & 0 & 22 & 0 \\ 0 & 40 & 32 & 22 & 0 & 35 \\ 20 & 20 & 35 & 0 & 35 & 0 \end{bmatrix}$$

The next largest element 40 is marked as zero and the corresponding edge (b, e) is removed as it is not a cut edge.



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 20 \\ 45 & 0 & 25 & 0 & 0 & 20 \\ 0 & 25 & 0 & 0 & 32 & \textcircled{0} \\ 0 & 0 & 55 & 0 & 22 & 0 \\ 0 & 40 & 32 & 22 & 0 & 0 \\ 20 & 20 & 35 & 0 & 35 & 0 \end{bmatrix}$$

The next largest element 35 is marked as zero and the corresponding edges (c, f) and (e, f) are removed as they are not the cut edges.



$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \boxed{20} \\ 45 & 0 & \boxed{25} & 0 & 0 & \boxed{20} \\ 0 & 25 & 0 & 0 & \boxed{32} & 0 \\ 0 & 0 & 55 & 0 & \boxed{22} & 0 \\ 0 & 40 & 32 & 22 & 0 & 0 \\ 20 & 20 & 35 & 0 & 35 & 0 \end{bmatrix}$$

Now the next largest element 32 is marked as the corresponding edge (c, d) is a cut edge and so on elements 25, 22 and 20 are also marked as their corresponding edges are the cut edges. Now the remaining edges form the minimal cost spanning trees with cost 32 + 25 + 22 + 20 + 20 = 119 and with path (a, f, b, c, e, d).

Conclusion

After examining the three basic different minimum spanning trees it can be stated that each technique has different idea to construct minimum spanning tree but all of them lead to the same results. Boruvka's and Kruskal's algorithms are clearly more useful if applied to the real world, while Prim's runtime grows far too quickly with the order of the graph to be of use in a serial processing environment. Also, we present a simple to apply technique to compute minimum spanning tree of undirected connected weighted graph using weight matrix and concept of cut edge. The technique can be easily applied to any length graph as decreasing order of weights is used.

References

[1]. Prim, R. C. (November 1957), "Shortest connection networks and some generalizations", Bell System Technical Journal 36 (6): 1389–1401

[2]. Dijkstra, E. W. (1959), "A note on two problems in connection with graphs" (PDF), Numerische Mathematik 1: 269–271,

[3]. Pettie, Seth; Ramachandran, Vijaya (2002), "An optimal minimum spanning tree algorithm", Journal of the ACM 49 (1): 16–34

[4]. Sunny Dagar "International Journal of Computer and Information Technology, Modified Prim's Algorithm"

[5]. Eisner, Jason. "State of the art algorithms for minimum spanning trees a tutorial discussion", (1997).

[6]. Dr.D Vijayalakshmir , R.Kalaivani (September 2014) "Minimum cost spanning tree using matrix algorithm", International Journal of Scientific and research publication, Volume 4, issue9.

[7]. Mandal, Dutta and Pal (October-2012) "A new efficient technique to construct A minimum spanning tree ", International Journal of Advance research in computer science and software engineering 2(10), , pp 93-97

[8]. M.R.Hassan , (2012) "An efficient method to solve least cost minimum spanning tree (LC-MST)problem". Journal of King saud University – Computer and information sciences 24,101-105

[9]. R. L. Graham and P. Hell, On the history of the minimum spanning tree problem, Ann. History Comput. 7 (1985), 43–57.

[10]. Joseph B Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. Proceedings of the American Mathematical society, 7(1):48–50, 1956.