© IJCIRAS | ISSN (O) - 2581-5334
January 2018 | Vol. 1 Issue. 8

MINIMUM SPANNING TREE ALGORITHMS AND TECHNIQUES

Sheikh Irfan Akbar[1], Shahid-ul-Islam [2]

[1] Computer Science and Technology, RIMT University, Mandi Gobindgarh, Punjab
[2] Information Technology, Desh Bhagat University, Mandi Gobindgarh, Punjab

Abstract

A spanning tree is a sub graph obtained from a connected graph which contains all the vertices of a graph. For a connected graph there may be many spanning trees. A minimum spanning tree of a weighted connected graph is the sub graph with minimum weight and no cycle. In this research, we have described the two well-known algorithms (prim's algorithm and kruskal's algorithm) to solve the minimum spanning tree problem. We have also described the applications, time complexity and comparison between the two algorithms. From the survey we have observed that prim's algorithm runs faster than kruskal's algorithm in dense graphs with maximum number of edges than vertices.

Keyword: spanning tree, minimum spanning tree, weighted connected graph, Prim's algorithm, Kruskal's algorithm, complexity.

1.INTRODUCTION

A spanning tree for a connected graph G is defined as the tree which contain all the vertices of graph G. there are many spanning trees for a single graph. A spanning tree is a connected sub graph with no circuits.
Example:

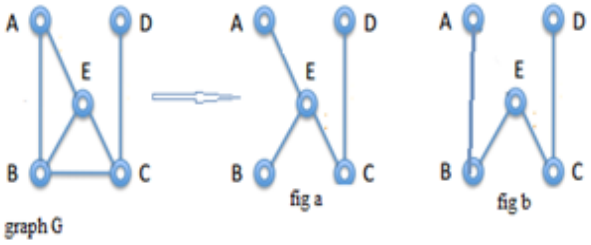


Figure 1 minimum spanning tree example

A minimum spanning tree of a weighted connected graph G (V, E) is the spanning tree which contains all the vertices with no cycles and the total weight of the edges of a tree is minimum. There can be multiple MST for a single graph but all these MSTs have a unique same weight.
In this paper we will discuss about the construction of MST from a weighted graph using different techniques or algorithms.

2. APPLICATIONS OF MST

I. **Network design:** The most important application of MST is in network design including telephone networks, TV cable network, computer network, road network, islands connection, pipeline network, electrical circuits, obtaining an independent set of circuit equations for an electrical network etc.
II. **Approximation algorithm for NP-hard problem:** this application for MST is used to solve travelling sales man problem.
III. **Cluster analysis:** k clustering problem can be viewed as finding an MST and deleting the k-1 edges, i.e. most expensive edges.
IV. **Indirect applications:** Apart from the above applications MST have also some indirect applications.
- Max bottleneck paths
- LDPC (low density parity check) codes for error correction
- reducing data storage in sequencing amino acids in a protein – image registration with Renyi entropy
- Model locality of particle interactions in turbulent fluid flow
- learning salient features for real-time face verification

3. MST ALGORITHM

There are various algorithms for minimum spanning trees. Prim's, kruskal's algorithm is the greedy algorithm which are used to find MST of an undirected weighted graph, which minimise weight and the number of edges.

3.1. PRIM'S ALGORITHM

The prim's algorithm was developed in 1930 by Jarnik and was later rediscovered by Robert C. Prim in 1957 and then again rediscovered by Edsger Dijkstra in 1959. It is also known as the DJP (Dijkstra-Jarnik Problem) algorithm, or the Jarnik algorithm, or the Prim–Jarnik algorithm.

The prim's algorithm is the greedy algorithm used to find the minimum cost spanning tree for an undirected weighted graph.

3.1.1. STEPS IN PRIM'S ALGORITHM

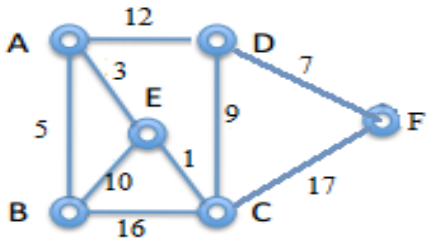
1. Initialise the tree with a single vertex from a given graph.
2. Choose the shortest outgoing edge from the node and add this to the tree.
3. The edge formed by the two vertices is treated as a single node, we check for the shortest adjacent edge to the node and add this to the tree.
4. Repeat the steps until all the vertices are added to the tree.

3.1.2. Pseudo code

Algorithm: PRIM (G)
Input: A weighted graph G
Output: A minimum spanning tree T for graph G.

1. Make a queue (Q) for all vertices of the graph G.
2. Set the priority to infinity to all the vertices except for the starting vertex whose priority is set to 0.
3. Pick any vertex v from G.
4. ~~←v~~
5. D[u] ~~←0~~
6. E[u] ← ∅
7. while Q ≠ ∅
8. u ← Q.removeMinElement()
9. add vertex u and edge (u,E[u]) to T
10. for each vertex z adjacent to u do
11. if z is in Q
12. if weight(u, z) < D[z] then
13. D[z] ← weight(u, z)
14. E[z] ← (u, z)
15. change the key of z in Q to D[z]
16. return tree T

Example:



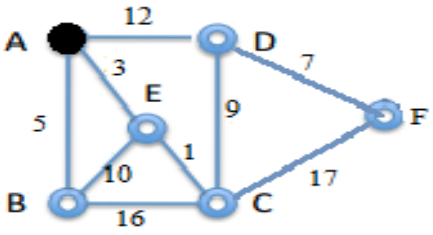
graph G

G= {V, E}

V= {A, B, C, D, E, F}

Step1: choose a source vertex A from the above graph.

Q = {B, C, D, E, F)



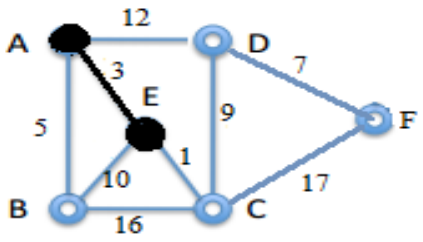
step1

Step2: vertex A is the source vertex v.

Adjacent vertices= {B, D, E}

Choose edge {A, E}

Q = {B, D, E, F)

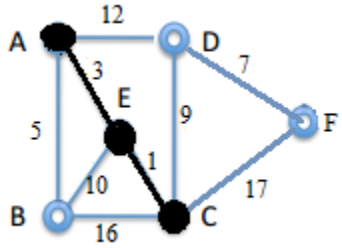


Step3:

Adjacent vertices= {B, D, C}

Shortest edge {E, C}

Q = {B, D, F)

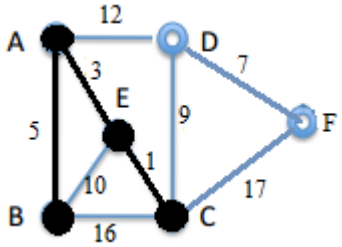


Step4:

Adjacent vertices= {B, D, F}

Shortest edge {A, B}

Q = {D, F)

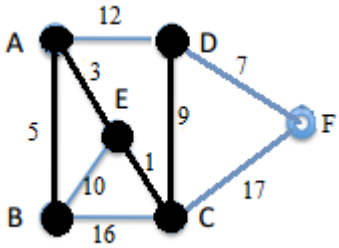


Step5:

Adjacent vertices= {D, F}

Shortest edge {C, D}

Q = {F}

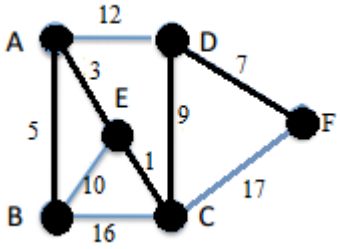


Step6:

Adjacent vertices= {F}

Shortest edge {D, F}

Q = {Ø}



Now we have the minimum spanning tree obtained from the graph G, which includes the edges {A, E}, {E, C}, {A, B}, {C, D}, {D, F}.

3.2. KRUSKAL's ALGORITHM

Kruskal's algorithm is named after Joseph Kruskal.it was developed in 1956. Kruskal's Algorithm is based on the greedy algorithm. The greedy method is applied as we pick the smallest weight edge that does not cause a cycle in the MST constructed so far. Kruskal's algorithm forms the forest where each edge acts vertex acts as a single tree, the edges are sorted in the increasing order of their costs and included in the set T of selected edges if the edges in the selected tree do not form a cycle and also after the inclusion of edges. Kruskal's algorithm also works on the disconnected graphs.

3.2.1. STEPS IN KRUSKAL'S ALGORITHM

1. Sort all the edges in the increasing order of their cost.
2. Choose the edge which has the smallest cost and check for the cycle. If the edge forms the cycle with

the spanning tree discard that edge else include the edge in the spanning tree.

3. Repeat step 2 until all the vertices are included in the spanning tree.

3.2.2. Pseudo code

MST-Kruskal (G, W)

Input: undirected graph G = (V, E)

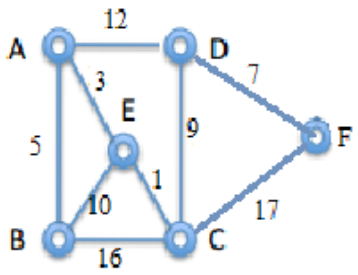
W = E ← R

Output: minimum spanning tree T

1. T = ∅;
   T (the final spanning tree) is defined to be the empty set;
2. For each vertex v of G do
3. Make empty set out of v.
4. Sort the edges of graph 'G' in the increasing order of their weight w.
5. For each edge (u, v) from the sorted edges do
6. If u and v belong to different sets then
7. Add (u,v) to T;
8. Union (u, v);
9. Return T;

EXAMPLE:

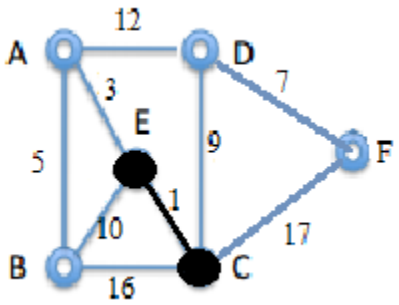
We have a graph G (V, E) shown in the fig below:



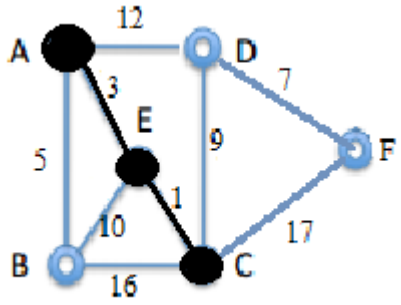
graph G

Set of edges	Weight (increasing order)	Edge included
E ——— C	1	yes
A ——— E	3	yes
A ——— B	5	yes
D ——— F	7	yes
C ——— D	9	yes
B ——— E	10	yes
A ——— D	12	No
B ——— C	16	No
C ——— F	17	No

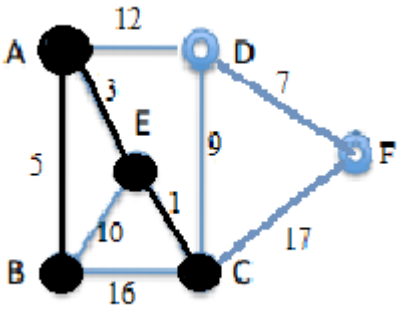
Step1: Include edge {E, C} which has lowest weight.



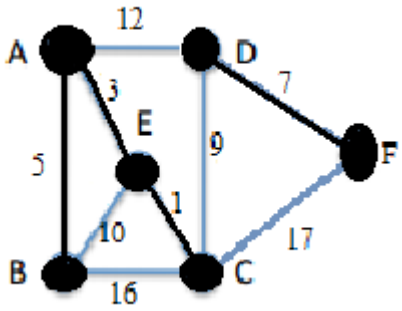
Step2: edge {A, E} is the next highest edge added to the tree.



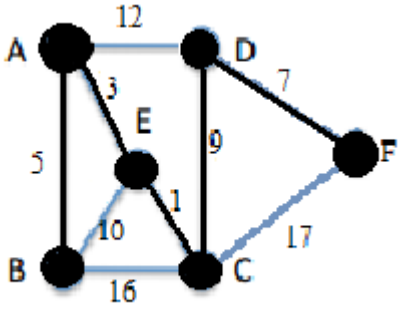
Step3: Edge {A, B} is added to the tree.



Step4: Edge {D, F} is added to the tree, which forms the disconnected tree.



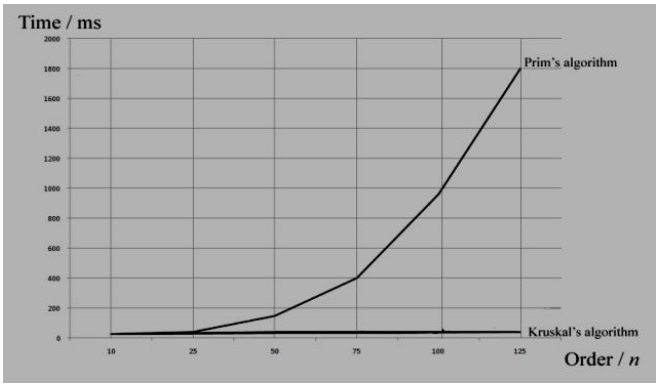
Step5: Next edge {D, C} is added to the tree, the edge joins the two trees.



All the vertices are included in the tree. This is the final minimum spanning tree which has the minimum weight among all the spanning trees. The edges {E, C}, {A, E}, {A, B}, {D, F}, {C, D}.

4. COMPARISON BETWEEN PRIM'S AND KRUSKAL'S ALGORITHM

1. Prim's algorithm begins with a node while as kruskal's algorithm starts with an edge.
2. Prim's algorithm works only on connected graphs, while as kruskal's algorithm can also be applied on disconnected graphs.
3. Prim's algorithm builds the minimum spanning tree by adding a vertex, while as kruskal's algorithm works on adding an edge at a time.
4. The time complexity of prim's algorithm (O(V2)) is less than that of kruskal's algorithm (O(logV)).



5. HISTORY AND FURTHER READINGS

Ronald Graham and Pavol Hell in the research paper describes the history of the problem up to 1985. The first known MST algorithm was proposed by Otakar Bor˚uvka in1926. He was a great Czech mathematician. He was considering an efficient electrical coverage of Bohemia at that time. In the mid-1950s when the computer age just began, the MST problem was further solved by many researchers. Among them, Joseph Kruskal and Robert Prim gave two commonly used algorithms. Both of them writes about Bor˚uvka's paper in their books. Prim's algorithm was a rediscovery of the algorithm by the prominent number theoretician Vojt˘ech Jarn´ık.

These algorithms run in O(m log n) time. Andrew Chi-Chih Yao, David Cheriton and Robert Tarjan

made improvements which reduces the time complexity to O(m log log n). With the invention of Fibonacci heaps, the complexity was further reduced. In the worst case, m = O(n) and the running time is O(m log∗ m). The complexity was further minimised to O(m log β(m, n)) by Harold N. Gabow, Thomas H. Spencer, and Robert Tarjan. David Karger, Philip Klein, and Robert Tarjan proposed a randomized linear-time algorithm for finding a minimum spanning tree in the restricted random-access model. Robert Tarjan gave a linear-time algorithm by using path compression. J´anos Koml´os showed that we can verify the minimum spanning tree in linear number of comparisons. Dixon, Monika Rauch, and Robert Tarjan gave the first linear-time verification algorithm .linear time algorithm for finding the minimum spanning tree remains an open problem.

6. CONCLUSION

In this paper we have discussed about well-known algorithms to construct and find the minimum spanning tree of an undirected and connected graph: Prim's algorithm, and Kruskal's algorithm. It is observed that the time complexity of prim's algorithm is less than kruskal's algorithm. The prim's algorithm is simple to implement. Prim's algorithm and kruskal's algorithms are mostly used for solving MST problem among the other algorithms. We are further looking to reduce the time complexity of MST algorithms.

REFERENCES

[1] Abhilasha.R, "Minimum Cost Spanning Tree Using Prim's Algorithm," International Journal of Advance Research in computer science and management studies, vol. 1, june 2013.

[2] D. K. M. Patel, Nimesh Patel, "A Survey on: Enhancement of Minimum Spanning Tree," Nimesh Patel International Journal of Engineering Research and Applications, vol. 5, january 2015.

[3] Jayanta Dutta. S.C.Pal, Ardhendu Mandal, "A New Efficient Technique to Construct a Minimum Spanning Tree," International Journal of Advanced Research in Computer Science and Software Engineering, oct, 2012.

[4] Neetu Rawat, "AN Algorithmic Approach To Graph Theory," in International Conference on Advances in Engineering & Technology,, Goa,INDIA, 2014.

[5] R.Kalaivani, Dr.D vijayalakshmir, "Minimum Cost Spanning Tree using Matrix Algorithm," International Journal of Scientific and Research Publications, vol. 4, september 2014.