COPYRIGHT © 2012 IJCIT, ISSN 2078-5828 (PRINT), ISSN 2218-5224 (ONLINE), VOLUME 03, ISSUE 02, MANUSCRIPT CODE: 130104

Modified Prim's Algorithm

Sunny Dagar

***Abstract*— This paper proposes a modified version of prim's algorithm which is a minimum spanning tree algorithm. Minimum spanning tree algorithms are greedy algorithms as they choose the best path available at that moment. Prim's algorithm chooses a root node randomly and starts processing it but choosing any node randomly as a root node is not efficient. So, in modified prim's algorithm, root node is chosen with minimum edge weight so that from the beginning of forest creation, only minimum weight edges are included. Minimum spanning tree is generated differently as of prim's algorithm. Although modified prim's algorithm is a special case of original prims algorithm with randomly chosen node is of minimum weight. With this modification in original prims algorithm, modified prim's algorithm maintains the complexity same as original prim's algorithm.**

***Index Terms*— minimum spanning tree algorithm, Modified prim's algorithm, greedy algorithm**.

I. INTRODUCTION

The term spanning tree is used when a node of a graph G is acyclic and capable of span the graph G. Minimum spanning trees are the trees with minimum cost of spanning the graph G. Many algorithms define the way to construct the minimum spanning tree. Most of these algorithms are greedy algorithms as at each step of algorithm only best available choices are made. Kruskal and prim algorithms are two main minimum spanning tree algorithms which are greedy as well and complexities of both the algorithms are same. [1, 2, 3]

Both kruskal and prim algorithm are efficiently produces the minimum spanning tree but in kruskal algorithm, disconnected components are generated after every step and these disconnected components will be connected at later stages. There is no continuous developing forest of connected nodes. In prim's algorithm although there is continuous developing forest of connected nodes but initially root node is chosen randomly which may or may not be the part of minimum edge of the graph. So in this modified prim's algorithm, along with continuous developing connected spanning tree, root node is also forming minimum edge of the graph. So in this way minimum edge is identified early and spanning tree is developed with minimum possible weight edges. [1, 2, 4]

II. LITERATURE REVIEW

Before explaining the working of proposed algorithm, some terminologies are there which are to be discussed.

Sunny Dagar is currently a student of Master of Technology program with specialization in Information Technology at Center for Development of Advanced Computing (CDAC), Noida, Uttar-Pradesh-201307 India (e-mail: dagarsunny08@live.com).

Safe Edge – Minimum spanning tree algorithms grows the tree one edge at a time with maintaining the loop invariant. Loop invariant prevents the formation of loop in the growing spanning tree by checking the formation of loop in the growing spanning tree by the newly added node and discards it if loop holds. [1, 2]

Cut – A cut (Z, Z-W) of an undirected graph G = (V, E) where V is a set of vertices and E is a set of edges of a graph G. An edge (x, y) belongs to E crosses the cut (Z, Z-W) if one of its end points is in Z and other is in Z-W. A cut respects a set X of edges if no edge of X crosses the cut. [1, 2]

Light Edge – A light edge is the edge in the graph G which is crossing the cut and its weight is minimum of all the edges crossing that cut. There may be more than one light edge in case when the weights of two edges are equal and they are satisfying the light edge property. E.g. – light edge in the diagram 1 is (c, f) and (c, d) with equal weights i.e. 8. So among these light edges algorithm can span any node randomly as these both are light edges. [1, 2]

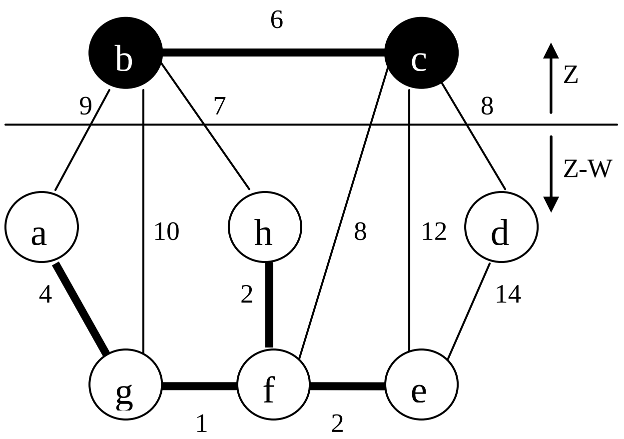
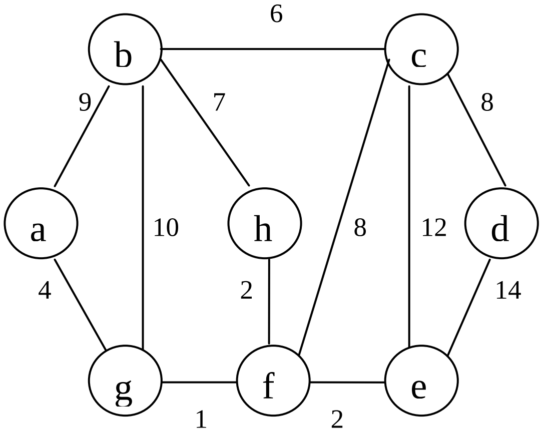


DIAGRAM: - 1

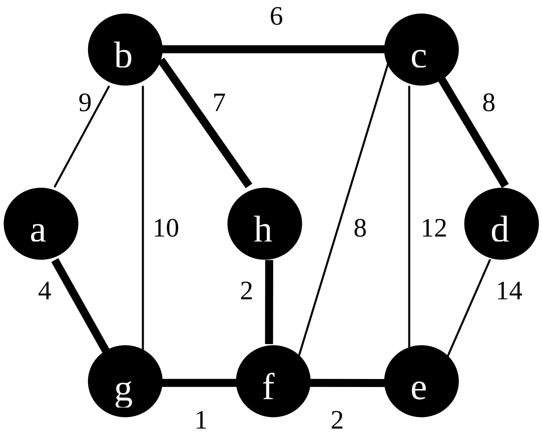
In diagram 1, there is an undirected graph G which is divided into two halves through a line. It is noted that, a graph can only be divided into two halves along un-spanned edges only. This line divides the graph into Z (upper half of the graph) and Z-W (lower half of the graph). The vertices in Z are shown in black and the vertices in Z-W are shown in white. The edges crossing the cut are the edges connecting the white vertices and black vertices. [1, 2]

Along with this explanation of these terms, we can understand clearly the meaning of minimum spanning tree through a diagram showing the undirected graph G and its corresponding minimum spanning tree. This diagram is capable of explaining the actual concept of minimum spanning tree.

1. Unconnected Graph G



2. Minimum spanning tree of Graph G



III. PROPOSED APPROACH

Algorithm:

MST-MODIFIED PRIM (*G, w, r*)

Here, *G* is an undirected graph, *w* is the weight of the edges of graph G and *r* is the root node.

Let 16 be infinity here.

V [G] denotes the set of vertices of graph G.

E [G] denotes the set of edges of graph G.

Jet [x] denotes the minimum weight of any edge connecting x to the tree.

P [x] denotes the parent of vertex x in the tree.

1. Repeat steps 1 to 3 for every x belongs to V [G]
2. Set Jet [x]:= 16 [Initialize every vertex to Infinity]
3. Set P [x]:= NIL [Initialize parent of every Node to NIL]
4. Extract minimum weight edge [r1, r2] from E [G]
5. Set Jet [r1]:= 0 [Set weight of root node 0]
6. Put this minimum edge [r1, r2] back in E [G]
7. Set Q:= V [G]
8. Repeat steps 8 to 13 while Q is not empty
9. Set x:= EXTRACT-MIN (Q) [Vertex with Minimum Jet value is extracted from Q and Put into x]
10. Repeat steps 10 to13 for each y belongs to Adj [x]
11. If y belongs to Q and *w*(x, y) < Jet [y]
12. Then Set P [y]:= x
13. Set Jet [y]:= *w*(x, y)

Performance of Modified Prim's algorithm:

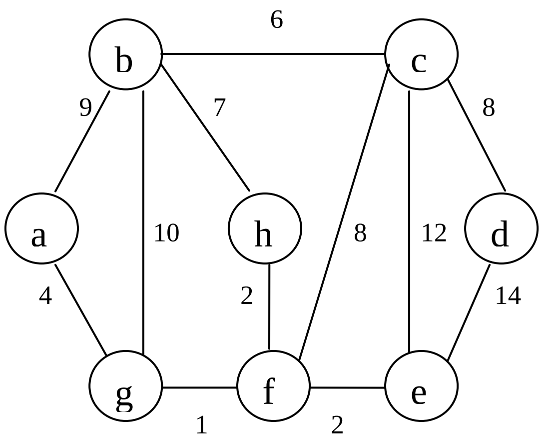
As like prims algorithm, the performance of modified prim's algorithm is based on the implementation of minimum priority queue Q. If we implement Q as a binary heap we can perform the initialization procedure in lines 1to 7 in O (V) time. Extract minimum operation in line 9 takes O (lg V) time and while loop from step 8 to 13 runs V times. So, Line 9 executes V times to make the total complexity O (V lg V). Assignment in step 13 takes O (lg V) time and for loop from step 10 to 13 repeats O (E) times which makes the total complexity of modified prim's algorithm O (E lg V + V lg V) which is equal to O (E lg V).

Performance comparison of minimum spanning tree algorithm:

	Kruskal Algorithm	**Prim's Algorithm**	**Modified Prim's Algorithm**
Complexity	O (E lg V)	O (E lg V)	O (E lg V)

<u>Example of Modified Prim's algorithm</u>:

Below is an unconnected graph G



<u>Step 1</u>: Jet value of all the vertices are initialized with infinity and parent value of every node is initialized as NIL.

Now minimum edge is extracted from E [G] i.e. (g, f).

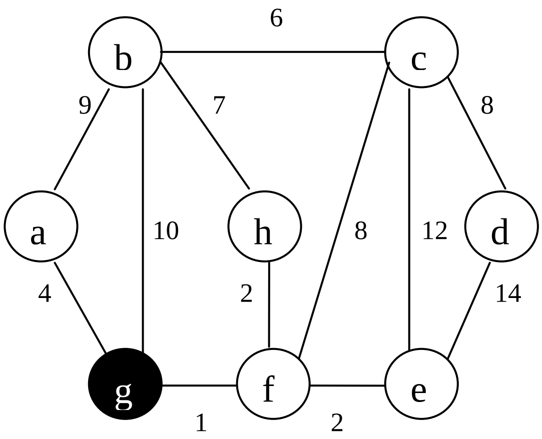
Now vertex g is set as s root vertex and its Jet [g] is initialized as 0.

Now put this edge i.e. (g, f) back to the set of edges E [G].

Copy all the vertices of V [G] into minimum priority queue Q.

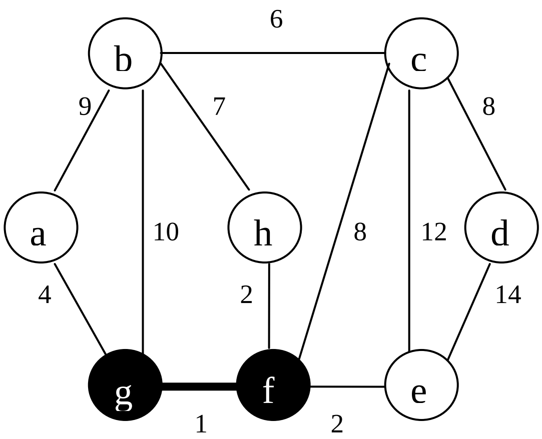
Now vertex g is extracted from Q as jet [g] is zero.

Now vertices that are adj [g] are extracted i.e. vertices a, b, f are extracted their Jet value is replaced with their respective edge weights. E.g. Jet [a] = 4, Jet [b] =10, Jet [f] = 1.

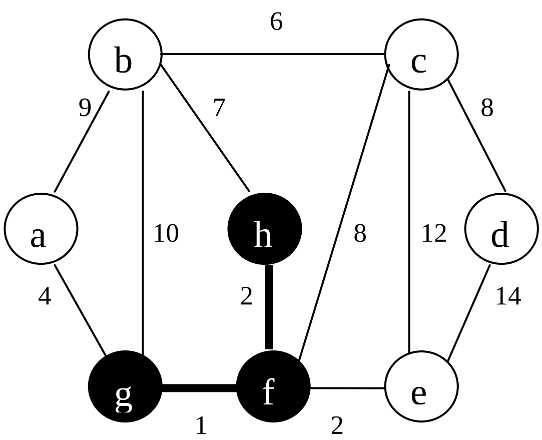


For rest of the steps only selected node is shown

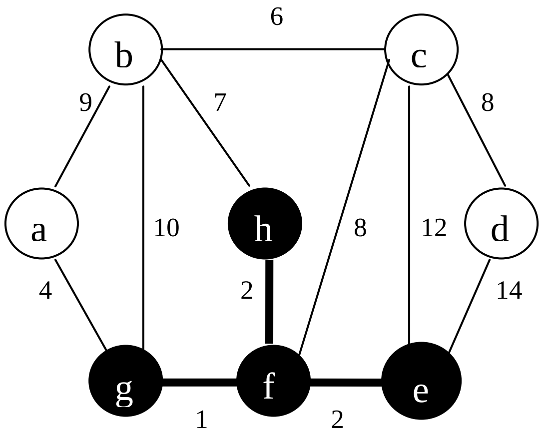
Step 2: Now selected node is f as edge (g, f) is light weight edge as well as safe edge.



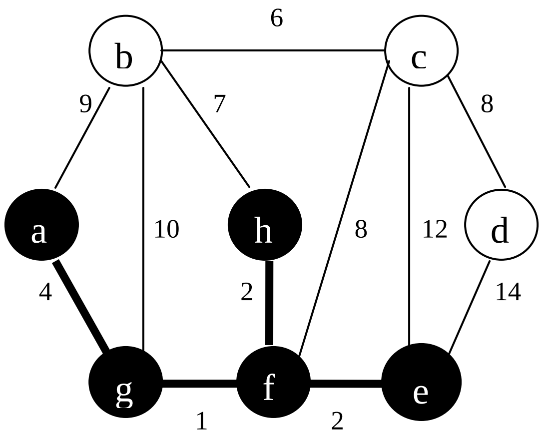
Step 3: Now selected node is h. Here we have two light weight edges i.e. (f, h) and (f, e) and both are safe edge as well so algorithm can select any of the edge. Here selected is (f, h) so node h is selected.



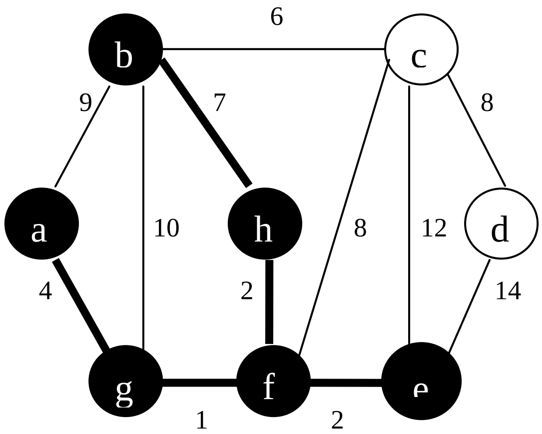
Step 4: Now selected node is e as edge (f, e) is light weight edge as well as safe edge.



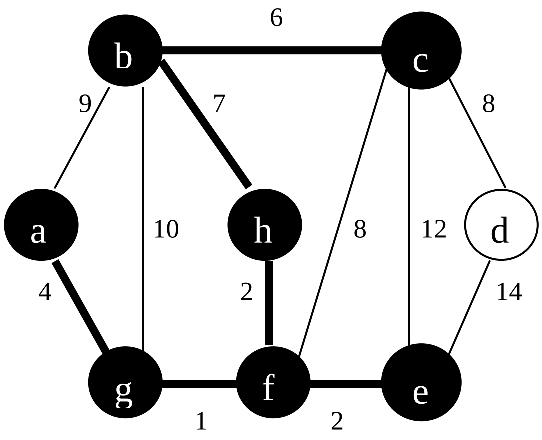
Step 5: Now selected node is a as edge (a, g) is light weight edge as well as safe edge.



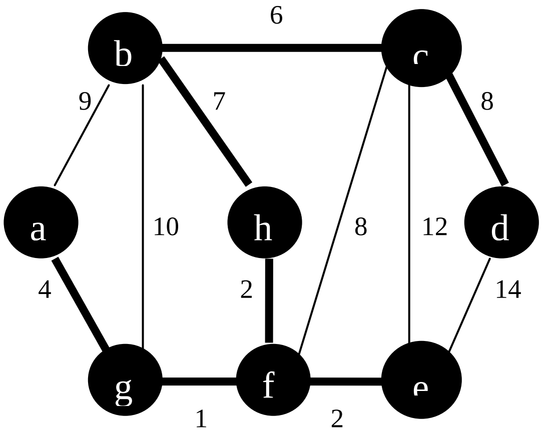
Step 6: Now selected node is b as edge (b, h) is light weight edge as well as safe edge.



Step 7: Now selected node is c as edge (b, c) is light weight edge as well as safe edge.



Step 7: Now selected node is d as edge (c, d) is light weight edge as well as safe edge.



This is our required minimum spanning tree.

IV. Conclusion

The proposed algorithm is slightly different from original prim's algorithm of minimum spanning tree. Its efficiency is same as original prim's algorithm. This modified prim's algorithm identifies the minimum weight edge of the graph and makes its one node as a root node. In original prim's algorithm, root node is randomly selected. So, modified prim's algorithm is a special case of original prim's algorithm in case randomly selected node is the node of minimum weight edge of the graph. As from starting of Modified Prim's algorithm vertex of minimum weight edge is selected, it gives slightly better performance in case where minimum weight edge is required from the starting phase of minimum spanning tree formation. So future work is mainly concern about reducing its complexity and makes it more efficient.

References

[1] Thomas H. Cormen, Charles E. Leiserson and Ronald L. Rivest, Introduction to algorithms, 2nd edition, prentice hall, 2002, ch. 23, pp. 561-579.

[2] Lecture 7: Minimum Spanning Trees and Prim's Algorithm. Available: http://www.cse.ust.hk/~dekai/271/notes/L07/L07.pdf

[3] Ekaterina Gonina and Laxmikant V, Parallel Prim's algorithm on dense graphs with a novel extension, 16th Nov, 2007. Available: http://www.eecs.berkeley.edu/~egonina/docs/PrimsAlgINTL07.pdf

[4] Arogundade O. T., Sobowale B., and Akinwale A. T, Prim Algorithm Approach to Improving Local Access Network in Rural Areas, International Journal of Computer Theory and Engineering, Vol. 3, No. 3, June 2011. Available: http://www.ijcte.org/papers/340-G469.pdf

Sunny Dagar is currently a student of Master of Technology program with specialization in Information Technology at Center for Development of Advanced Computing (CDAC), Noida, Uttar-Pradesh-201307 India. Prior to admission in CDAC, he completed his Bachelor of Technology program with specialization in Computer Engineering in 2011 from Rajasthan Technical University, Kota, Rajasthan-324010 India.

His research interests include algorithms, cryptography and steganography.