Visualization of Prim's and Kruskal's MST Algorithms

Goutham Nerevetla & Jeremiah White

April 20, 2018

1 Background and Related Work

Prim's and Kruskal's algorithms are two common solutions to finding the Minimum Spanning Tree (MST) in a connected graph. The MST is defined as the spanning tree of a given graph such that the sum of the edges is minimum. In this project we implemented both Prim's and Kruskal's algorithms with a visualization of each at every iteration of the algorithm. In completing this project we have gained a better understanding of each algorithm. The visualization aspect of the project serves to give a clearer picture of the process by which each algorithm finds the MST, making learning the algorithm simpler.

1.1 Prim's MST

Prim's MST takes a graph and position as input; it prints and the returns the MST. Prim's finds the MST by finding the smallest weight edge such that u is a node in the MST and v is not [2]. Prim's runs V times where V is the number of nodes in the graph.

```
function PRIM(G, pos)
    $subG \leftarrow copy\ of\ G\ with\ only\ the\ start\ node$
    for i in range 1 to number of nodes in G do
        $edge \leftarrow MINIMUMDISTANCE(G, subG)$
        $subG \leftarrow edge\ with\ color = 'g'$
        $Draw\ at\ pos\ and\ show\ the\ Graph$
    end for
    return $subG$
end function

function MINIMUMDISTANCE(G, subG)
    $minKey \leftarrow largest\ edge$
    for u,v in G do
        if node(u) in subG & node(v) not in subG & G[u][v]['weight'] < minKey['weight'] then
            $minKey \leftarrow G[u][v]$
        end if
    end for
    return $minKey$
end function
```

1.2 Kruskal's MST

Kruskal's MST takes a graph and position as input: it prints and returns the MST. Kruskal's finds the MST by creating subsets consisting of each node in the graph, then sorting all of the edges in the graph by weight. Then starting at the lowest weight edge, add the edge to the MST if the edge connects two subsets [3]. The subset of the graph are created using the union-find data structure. A union-find data structure allows us to create a grouping of disjointed sets from the nodes in the graph, and quickly find if elements are in the same set [1]. We can then merge sets when edges are added to the MST.

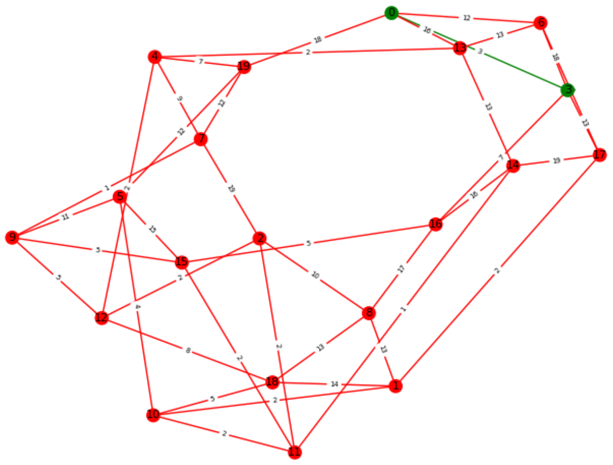
```
function KRUSKAL(G, pos)
    $subG \leftarrow empty\ graph$
    $subSets \leftarrow UnionFind()$
    $EdgeList \leftarrow sort(G)$
    $i \leftarrow 0$
    while selected edges < number of edges in G do
        $edge \leftarrow Edgelist[i]$
        if subsets[edge[0]] not in subsets[edge[1]] then
            $union(edge[0], edge[1])$
            $subG \leftarrow edge\ with\ color = 'g'$
            $Draw\ at\ pos\ and\ show\ the\ Graph$
        end if
        $i++$
    end while
    return $subG$
end function
```

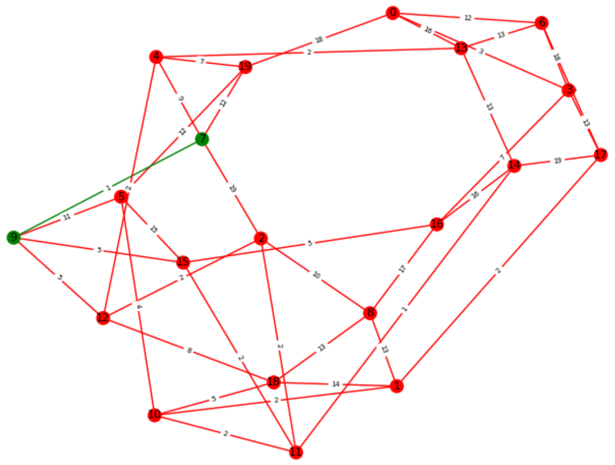
2 Setup

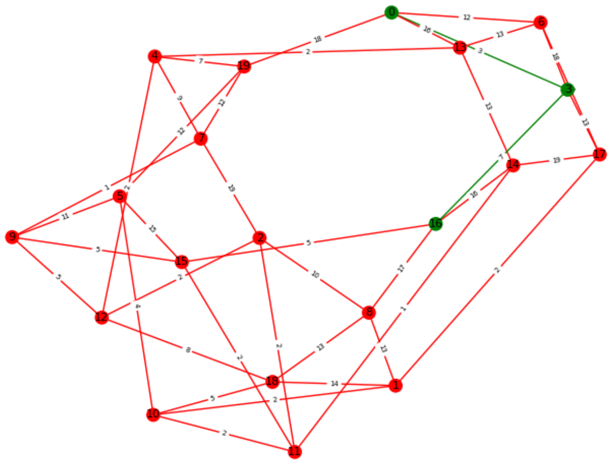
For this project we used Python 3.6, Developing using the PyCharm IDE. We also used the NetworkX version 2.1 Python package to represent our graph data structures. We used matplotlib to draw our graphs.

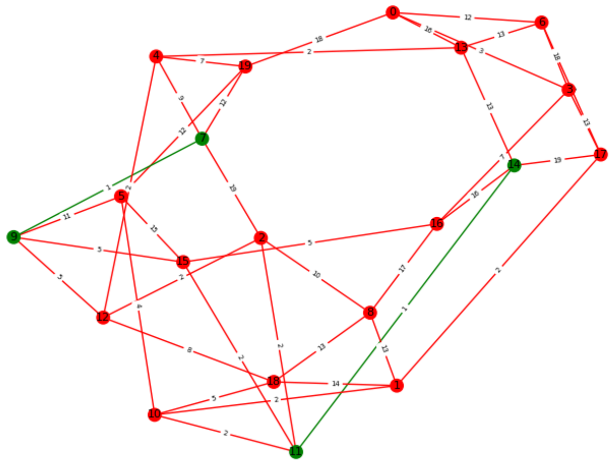
3 Results

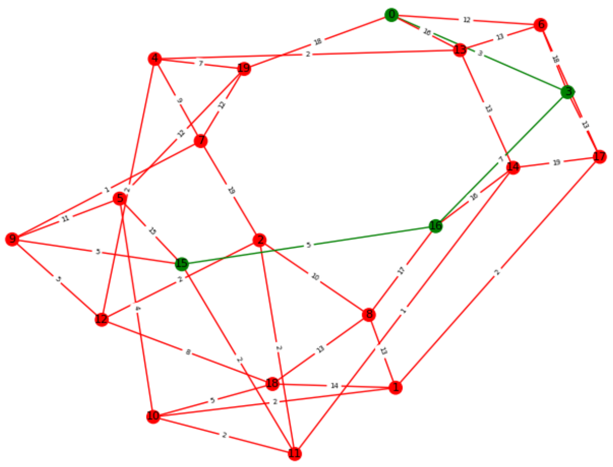
We will look at how Prim and Kruskal each select edges to be added to the MST by examining the first three iterations of each on a randomly generated graph with 20 nodes. With Prim's being on the left and Kruskal's on the right

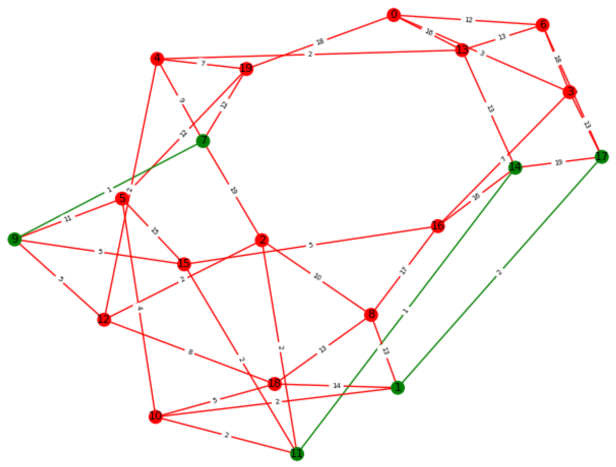












4 Conclusions

As we can see from the six images above the two algorithms find the MST in very different ways. Prim's on the first iteration starting from node 0 selects the edge (0, 3, 3) and adds it to the MST. On the second iteration selects the (3, 16, 7) edge. On the third iteration selects the (16, 15, 5) edge. Kruskal's on the other hand first selects the (7, 9, 1) edge. On the second iteration selects the (11, 14, 1) edge. On the third iteration selects the (1, 17, 2) edge. From the graph visualization it becomes clear the process each algorithm take to find the MST. Prim's is only selecting the smallest edge which is connected to nodes in the MST. On the other hand Kruskal's is starting with the smallest weight edges and combining if the nodes are in separate sets. Prim's appears to grow out form the start node, while Kruskal's combines the separate sets over time. We believe this project will be useful for any person that is interested in learning how these algorithms work.

References

[1] Bernard A. Galler and Michael J. Fisher. 1964. An improved equivalence algorithm. Commun. ACM 7, 5 (May 1964), 301-303. DOI=http://dx.doi.org/10.1145/364099.364331

[2] R. C. Prim 1957. Shortest connection networks and some generalizations. The Bell System Technical Journal Volume: 36, Issue: 6,(Nov. 1957) DOI=10.1002/j.1538-7305.1957.tb01515.x

[3] Joseph B. Kruskal 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. Amer. Math. Soc. 7 (1956), 48-50. DOI=https://doi.org/10.1090/S0002-9939-1956-0078686-7