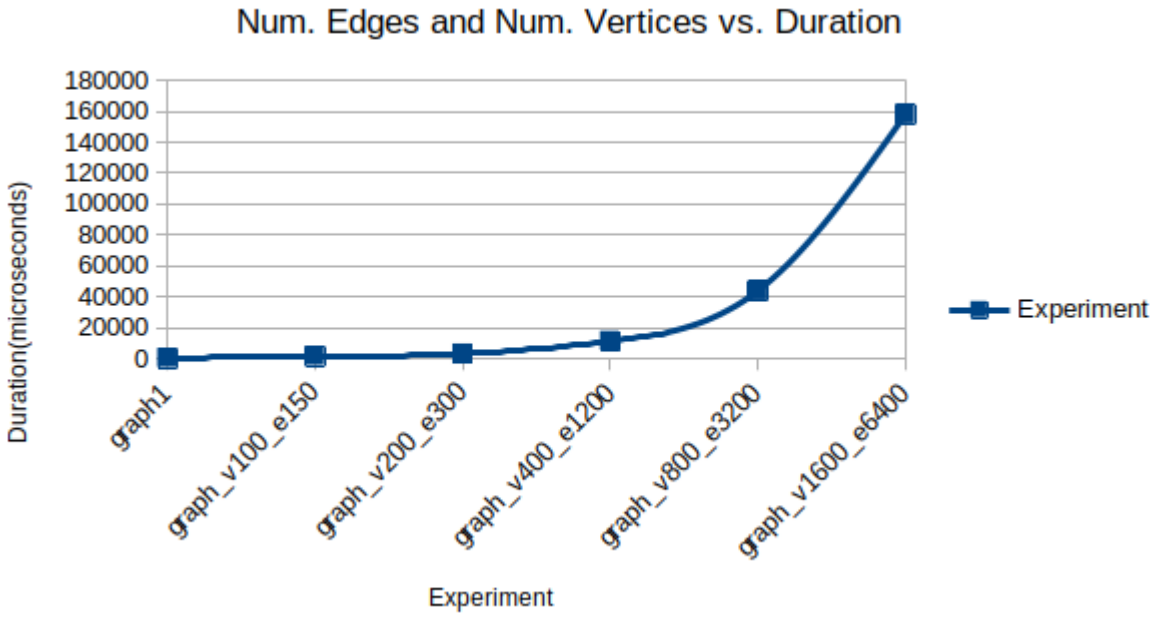
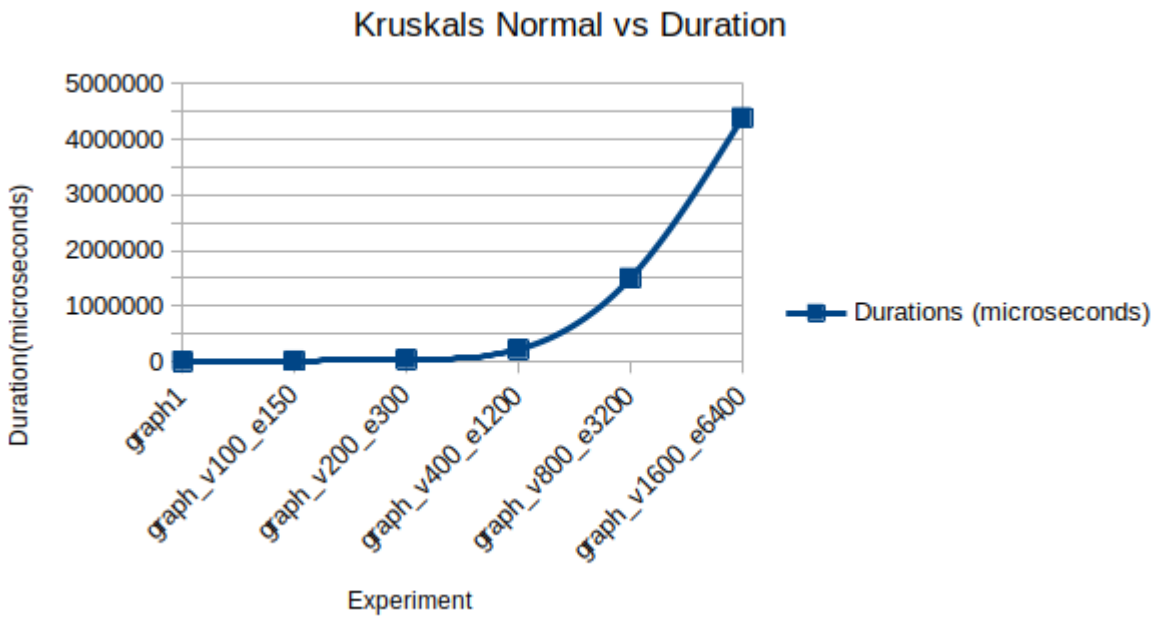
Wyatt Shinkle
Algorithms HW5 Writeup

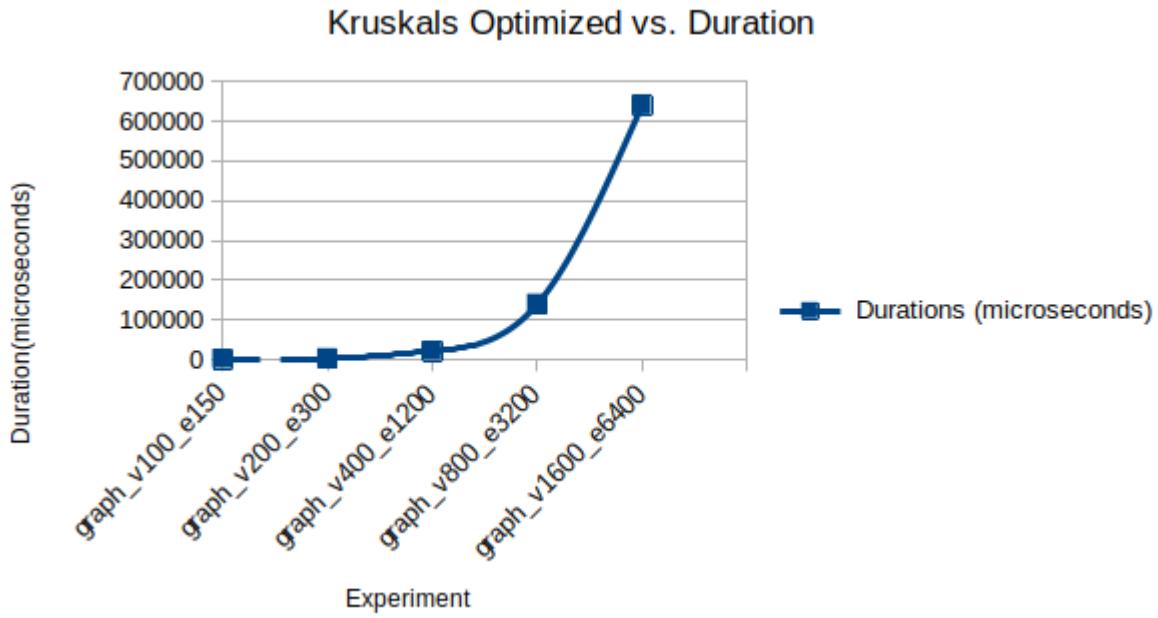
1. I implemented Prim's Algorithm by first loading the list of vertices and weights into an adjacency matrix which I thought would make it easier to later use these values. Next I initialized a few different vectors. One holds the key values, one holds all the parents, and another holds a list of boolean values that represents set of vertices including in MST. After this I set the first value of the key vector to 0 and the first value of the parent vector to -1. Then for each vertices I find the the index of the minimum – key or weight. Then set the value of the mstSet vector to true because it has been determined. After this another for loop checks if the lowest values has been found and adds it to the parent and key vectors. Finally, I add each edge to a vector holding the mst.



This chart describes how long each experiment took to run with varying numbers of edges and vertices. This kind of growth roughly matches the time complexity of Prims Algorithm which is O(ElogV).

2. I first implemented Kruskals Algorithm using STL's sets. After loading the list of vertices. I sorted the vector of edges in increasing order based on their weights. After this you simply go down each entry in this vector and set the parent of one of the vertices to the value of the other vertex. Each time you do this you need to check if adding an edge with form a cycle which we do by seeing if the two vertices are within the same set. You know the process is down when every vertex is in the same set. Implementing Kruskals Algorithm with Data Compression is somewhat similar but now whenever the find function is called it will flatten the tree of vertices so that indexing more recently used vertices is faster.





Both of these also roughly match the time complexity of both of their algorithms with is roughly O(ElogV). Out of all three algorithms Prims Algorithm greatly outperformed Kruskals. Along with this optimizing Kruskals did not seem to have a huge affect but this may be due to the fact that I used the STL sets in order to implement the normal version which may have increased performance.