Assignment Question 7: Analysis of Algorithms **Jonathan Taylor**
COMS2004 1665909
Dr. Ritesh Ajoodha Due Date: 23 October 2018

Problem Statement

This researcher was tasked with researching two algorithms that are commonly used to extract the *Minimum Weighted Spanning Tree* (MWST) from a graph G, namely Prim's and Kruskal's algorithm.

Hypothesis

It is predicted that Kruskal's and Prim's algorithm will perform similarly, but in can outperform one another in different scenarios.

Rationale

Each algorithm has different parameters that will allow one to outperform the other. For example, Kruskal's algorithm will perform better with sparse graphs as it is often implemeted with simpler data structures, while Prim's algorithm will perform better with dense graphs with more edges than vertices.

Motivation

Both Prim's and Kruskal's algorithm are generally used in the design of efficient networks and routes. An example of this would be to connect telephones together in a telephone network in the most efficient way. Another useful application would be to solve the classic traveling salesman problem, where one tries to work out the most efficient way for a person to visit many locations without visiting the same location more than once, i.e. no loops.

Literature review

Prim's algorithm was developed in 1930 by Czech mathematician Vojtch Jarnk and later re-discovered by computer scientists Robert C. Prim in 1957 and Edsger W. Dijkstra in 1959. Kruskal's algorithm was developed by Joseph Kruskal and first appeared in Proceedings of the American Mathematical Society, pp. 4850 in 1956. Other algorithms for this problem include the Reverse-delete and Borvka's algorithm.

Analysis & Testing

Prim's algorithm results in a MWST, a minimum weight connected graph with no cycles. Prim's algorithm will start with an empty list called "visited", which is used to keep track of vertices the algorithm has visited. The algorithm then picks an arbitrary vertex, adds it to visited and then examines all vertices reachable from this first vertex. The algorithm will then pick the edge with the smallest cost which leads to an unvisited vertex. Now Prim's will examine all vertices in the graph reachable from these two vertices in visited and again choose the edge with the smallest cost which leads to an unvisited vertex. This process will be repeated until

all vertices have been visited.

Kruskal's algorithm also results in a MWST. Kruskal's algorithm starts examining the graph and choosing the edge with the smallest cost. The algorithm will then repeatedly look for the smallest edge in the entire graph which will not create a cycle. This will be done until all vertices have been visited. However, unlike Prim's algorithm, this could result in many smaller subgraphs of G if the MWST is not connected, this result is called a *minimum spanning forest*.

Both algorithms can be shown to run in $\mathcal{O}(|E| \log |V|)$, where E is the number of edges and V is the number of vertices in a graph. Prim's algorithm can be reduced to $\mathcal{O}(|E| + |V| \log |V|)$, which is useful for graph with many edges, where the algorithm's performance will tend to linear time, $\mathcal{O}(|E|)$, with the use of a Fibonacci heap and adjacency list. Prim's algorithm outperforms Kruskal's algorithm when the graph in question has many more edges than vertices, known as a 'dense' graph. Kruskal's algorithm can be reduced to $\mathcal{O}(|E|\alpha(|V|))$, if the edges are sorted or can be sorted in linear time and a disjoint-set data structure is used. This is an desirable running time as $\alpha(|V|)$ is an extremely slow growing function, being the inverse of the *Ackermann function*. Kruskal's algorithm generally performs Prim's algorithm in cases where the number of edges and vertices are the same, known as a 'sparse' graph.

Final Evaluation

This researcher was tasked with researching two algorithms that are commonly used to extract the *Minimum Weighted Spanning Tree* (MWST) from a graph G, namely Prim's and Kruskal's algorithm. It was found that both algorithms can be shown to run in $\mathcal{O}(|E| \log |V|)$, and could be improved using more complex data structures such as adjacency lists. Furthermore, Prim's algorithm outperforms Kruskal's algorithm when the graph in question has many more edges than vertices, which is known as a 'dense' graph.