Matthew Roknich, A11323601
Horea Porutiu, A97068584

CSE 101 – PA2

In Problem 4, I was tasked with modifying my Dijkstra's algorithm to incorporate elements of Prim's algorithm. This Dijkstra/Prim hybrid incorporates a conditional weight that represents the degree to which existing source to vertex path length in the growing tree.

To illustrate the point… let's take a look at a code segment from PrimDijk.cpp

```
float alt = c * g.vertices[u.first]->distance + g.get_weight(u.first,*it);
if (alt < g.vertices[*it]->distance && g.vertices[*it]->visited != true) {
…
```

Using a conditional weight, we can effectively create a gradient between the SPT of Dijkstra and MST of Prim. For a value of C = 0, we will get Prim's algorithm. For a value of C = 1, we will get Dijkstra's algorithm.

As a result, we can input values gradually increasing from Prim's, eventually reaching Dijkstra. In the following chart, I calculated the cost of a randomly generated graph of 5000 nodes with varying conditional weights and probabilities.

	P = 0.1	0.3	0.5	0.7	0.9
C = 0.0	5106.37	5035.09	5023.08	5014.91	5010.75
0.1	5106.54	5036.1	5026.28	5018.23	5014.85
0.2	5108.09	5043.14	5042.33	5040.39	5046.24
0.3	5114.87	5079.21	5111.24	5130.48	5143.98
0.4	5139.99	5196.82	5253.12	5262.09	5275.43
0.5	5262.1	5358.38	5404.08	5400.99	5376.17
0.6	5491.95	5553.18	5572.64	5507.56	5446.97
0.7	5779.82	5796.33	5737.59	5586.11	5508.93
0.8	6058.55	6017.38	5875.88	5655.26	5586.15
0.9	6363.79	6217.16	5976.29	5731.53	5662.65
1.0	6678.32	6394.13	6073.34	5813.62	5737.54

Most importantly, when I graph the values recorded in the table, it gives some interesting implications about the relationship between sparse/dense graphs and the two algorithms.

As we can see in the figure below, the algorithm had the highest weight for a sparse graph with a conditional constant of 1.

In other words, we can make some conclusions about the best times to use Prim's algorithm and Dijkstra's algorithm. When you have a sparse graph, it would be more efficient to use Prim's algorithm. When you have a dense graph, it would be more efficient to use Dijkstra's algorithm.