CS 374 **Spring 2015**

Homework 5

Nikhil Unni (nunni2)

1.

sDist ← Dijkstra(s, V, E);
eReverse ← $\emptyset$;
foreach $(u, v) \in E$ **do**
  eReverse ← eReverse $\cup (v, u)$
end
tDist ← Dijkstra(t, V, eReverse);
out ← $\emptyset$;
minDist ← ∞;
foreach $e = (u, v) \in E'$ **do**
  if $out = \emptyset$ *or* $sDist(s, u) + l(e) + tDist(t, v) < minDist$ **then**
    out ← e;
    minDist ← $sDist(s, u) + l(e) + tDist(t, v)$;
  end
end
output out;

The algorithm is a modification on Dikstra's algorithm. I find the shortest paths from s to all other points, and the shortest paths from all other points to t by reversing the edges in the graph and running Dijkstra's algorithm on the reversed graph starting from t.

Then, the shortest additional edge is just the edge $e = (u, v)$ with the shortest distance from s to u, plus the length of e, plus the shortest distance from v to t.

We can prove that this is indeed the shortest path by contradiction. Assume that the sum of these 3 is not the shortest path from s to t through e. Because we have to go through e, now the path from s to u is some path that's not the shortest path. By definition this path must be greater than or equal to in length of the shortest path, which can only increase the path length. Therefore, the sum above is the shortest, or a tie for the shortest path.

The algorithm runs in O(m + nlogn + k) time because we do Dikjstra's algorithm a constant 2 number of times, and then we iterate through the entire list of E'.