Eric Yang

304263623

CS 180

Dis 1B

Homework 6

1.

2. A. Since the lexicographically minimum tree has the smallest possible tuple out of all spanning trees in which every edge is the smallest possible, that would mean the sum of all the edges must be the smallest possible; therefore, it is a minimum spanning tree.

B. We can recursively choose a value x that is between the largest weight and smallest weight of edges. Then we can perform a breadth first search starting from u and check if we can reach v using only edges with weight less than or equal to x. If this is possible, repeat process choosing a value between smallest weight and x. If this is not possible, repeat process choosing a value between x and biggest weight. The path with smallest maximum weight will be the path with maximum weight x when there is only 1 value left. This solution should run in O(nlogn) time and correct because the algorithm will attempt to reach v from u while only taking paths weighing <= x; once x is the minimum possible that still has a path from u to v, we have the correct solution.

3. A. Proof by contradiction: we have two minimal spanning trees T and T’ of the graph. Let e be an edge of minimum weight from x to y in T but not in T’. Since e is not in T’, then weight of e must be > weight of every edge in T’ in path from x to y. However, since e is edge with minimum weight in T, every edge in the path in T’ from x to y is also in T. But this forms a cycle and contradicts T being a tree.

B. We can find all minimum spanning trees by running Prim’s algorithm normally on the graph. However, when we reach points where there are more than one edge with the lowest weight that can be added, we must branch off and recursively call Prim’s algorithm again on each edge. Possibly choosing different edges will lead to different minimum trees.

4. Consider the parabola y = x^2 and set of points (x1,x1^2)...(xn, xn^2). We see that the convex hull requires the sorted order of x1…xn as we traverse along the parabola. Thus convex hull can’t be faster than sorting the points. The sorting of points will have complexity O(nlogn) since we will be using comparison based sorting similar to mergesort or quicksort. Thus convex hull requires at least O(nlogn) as well.