The run time of this algorithm is O(n^2). To show why this is I have split the maxSpanningTree method in to 4 parts where E = number of edges and N = number of node:

For getting all the edges edges, its O(2E) because each edge is represented twice once going each way, and it iterates through all edges.

Java sort method I am using Collections.sort() to sort the list of edges. This has worst case of O(n) because it uses a modified version of merge sort.

Main loop is E \* (N\*2E), because it iterates through all edges and in each iteration it calls Graph.isPath which is reclusive and worst case it will iterate though all nodes in the graph and all edges making it O(N\*2E).

Printing is N\*2E for the same reason Graph.isPath is because they yser a very similar method of traversing the graph. But here E is the number of edges in the Max spanning tree not in the whole graph.