MAT-72006 Advanced Algorithms and Data Structures
November 17, 2016
HW 9: 23 Minimum Spanning Trees, 24 Single-Source Shortest Paths

1.

Prove or disprove the following claim. Let G be a connected graph. Then, G has a spanning tree that is a binary tree.

2.

A *complete graph* on n vertices, denoted by K_n, is the graph on n vertices having all possible $\binom{n}{2}$ edges. Prove or disprove the following claim. For every $n \geq 2$, every spanning tree T of K_n has the property that any two vertices of T are at a distance at most 2 from each other.

3.

Let G be an edge-weighted graph. Prove that if all edge weights of G are positive, then any subset of edges that connects all vertices and has minimum total weight must be a tree. Also, prove that the same claim does *not* hold if we allow some weights of G to be nonpositive.

4.

For this question, all graphs are unweighted, undirected, and connected with no parallel edges nor self-loops (i.e., simple graphs). Also, every path is simple, i.e., contains no repeated vertices. Using say breadth-first search we can determine the shortest path distance between two distinct vertices s and t in a graph G. But just how many shortest paths can there be between s and t, and how does this affect possible algorithms for related problems? In particular, consider the following.

(a) Show that there is an infinite number of graphs with the following properties:

- there is exactly one shortest path between s and t,
- when the shortest path distance between s and t is d, there are no two vertices s' and t' distinct from s and t, respectively, such that the distance $d(s', t') > d$, and
- the graph is not a simple path ○—○—○— ··· —○ itself.

(b) Show that there is an infinite number of graphs with the following properties:

- there is at least one pair of distinct vertices s and t such that there is exactly one shortest path between s and t, and
- the number of simple paths between s and t is at least 16.

(c) Show that there is an infinite number of graphs with the following property:

- the number of shortest paths between s and t is exponential in V, i.e., the number of vertices. (Hint: for example, come up with an n-vertex graph such that there are $2^{\Theta(n)}$ shortest paths between two vertices s and t).

5.

Based on the previous problem, there are graphs that can have many (shortest) paths between two vertices. Despite this fact, there are efficient algorithms for finding a (shortest) path. Do you find this surprising?

(a) Can you name another problem with similar properties, that is, the number of possible solutions is large (i.e., exponential or non-polynomial in the input size) yet there is a polynomial-time algorithm for the problem?

(b) Finally, consider the problem of enumerating (e.g., listing all) shortest paths between two given vertices s and t in a graph. Is there an efficient, i.e., a polynomial-time algorithm for the problem?