Algorithm Design and Analysis

Assignment 2

Deadline: Nov 5, 2024

1. (25 points) Given a direct graph $G = (V, E)$ and two vertices $s, t \in V$, design a polynomial-time algorithm to decide if we can go from s to t such that each vertex is visited at least once. Prove that your algorithm is correct, and analyze its running time.

2. (25 points) Given an undirected connected graph $G = (V, E)$ and a vertex s, prove that the DFS and BFS trees rooted at s are identical if and only if G is a tree.

3. (25 points) It is possible that the shortest path from s to t in an edge-weighted graph is not unique. Given a directed edge-weighted graph $G = (V, E, w)$ with positive edge weights and $s \in V$, design an $O((|V| + |E|) \log |V|)$ time algorithm to output a Boolean array B, with vertices being the array indices, such that $B[u] =$ `true` if the shortest path from s to u is unique and $B[u] =$ `false` otherwise. You can assume that every vertex $u \in V$ is reachable from s. Prove the correctness of your algorithm.

4. (25 points) Consider a directed *acyclic* edge-weighted graph $G = (V, E, w)$ where edge weights can be negative and a vertex $s \in V$.

   (a) (20 points) Adapt the Bellman-Ford algorithm to find the distances of all vertices from s. Your algorithm must run in $O(|V| + |E|)$ time. Prove the correctness of your algorithm and analyze its time complexity.

   (b) (5 points) Suppose now we want to find the length of the longest path from s to each vertex $u \in V$ (the length is measured by the sum of the weights of the edges on the path, not by the number of the edges). Can we negate the weight of every edge and use the algorithm from the first part? If so, prove it; if not, provide a counterexample.

5. How long does it take you to finish the assignment (including thinking and discussion)? Give a score (1,2,3,4,5) to the difficulty. Do you have any collaborators? Please write down their names here.