Homework No. 5

CSCI 4470/6470 Algorithms, CS@UGA, Spring 2019

Due Thursday April 18, 2019

There are six questions with 120 points in total. Q6 is a bonus question for CSCI 4470 students.

1. (15 points) Show how the algorithm RECURSIVE DFS (along with TO-START-DFS) in the Lecture Note 4 works on the graph figure shown below. Assume for the graph, the adjacency list of every vertex is ordered alphabetically. Assume that the for loop of lines 4-5 of algorithm TO-START-DFS considers the vertices in the alphabetical order. Show the resulted depth-first-search tree (or forest).

2. (25 points) Consider the following Constrained Reachability problem: the input consists of a directed graph $G(V, E)$, two vertices $s, t \in V$, and a obstacle subset $U \subseteq V$, where $U \cap \{s, t\} = \emptyset$. The output is “yes” if and only if there is a path from $s$ to $t$ in the graph without using any vertex in $U$. Apparently, the problem is the commonly known REACHABILITY problem when $U = \emptyset$.

(1) Design a recursive DFS type of algorithm to Constrained Reachability solve problem. You may assume that the input graph is already stored in adjacency lists and variable $v.o$ indicates if vertex $v$ is in the
subset $U$ or not. The information about $v.o$, for every vertex $v$ is also given along with the graph.

(2) Argue that your algorithm still runs in linear time (i.e., $O(|E| + |V|)$-time) though the subset $U$ may contain a large number of vertices.

3. (20 points) It is known that depth-first-search strategy can be used to determine if a given undirected graph contains a cycle. Argue that the algorithm runs in time $O(|V|)$, independent of $|E|$ before it answers the question correctly. Note that you need to prove that time $O(|V|)$ is enough for the algorithm to answer both “yes” and ”no”.

4. (25 points) (1) Run the Bellman-Ford algorithm on the directed graph of Figure 1, using vertex 4 as the source. Show $d$ and $\pi$ values after each of the $|V| - 1$ passes.

(2) Modify the graph so that it contains at least one negative cycle. Run the Bellman-Ford algorithm again to show how the negative cycle can be detected by the algorithm.

```
Figure 1: Graph for Question 4.
```

5. (15 points) Give a simple example of a directed graph with negative weight edges for which Dijkstra’s algorithm produces incorrect answers. Also argue that, if all negative weights occur only on the edges leaving the source $s$, Dijkstra’s algorithm remains correct.

6. (20 points, Graduate student only, bonus for undergraduates)
How do we use Floyd-Warshall algorithm to detect if the input graph contains a negative weight cycle?

The answers must be word-processed or typed. You may substitute formulae and figures with hand-writings. Your submitted algorithms should
be in the pseudo-code, not in any specific programming language. Answers deviating from these requirements will be returned without grading.

The answers must be the student’s own work. Idea sharing and referencing to others’ work are not allowed. Plagiarism and other forms of academic dishonesty will be handled within the guidelines of the Student Handbook and reported to the University.