COMPSCI330 Design and Analysis of Algorithms
Assignment 5

Due Date: Wednesday, October 25, 2017

Guidelines

- **Describing Algorithms** If you are asked to provide an algorithm, you should clearly define each step of the procedure, and then analyze its overall running time. There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice. If the running time of your algorithm is worse than the suggested running time, you might receive partial credits.
- **Typesetting and Submission** Please submit each problem as an *individual* pdf file for the correct problem on Sakai. LaTeX is preferred, but answers typed with other software and converted to pdf is also accepted. Please make sure you submit to the correct problem, and your file can be opened by standard pdf reader. **Handwritten answers or pdf files that cannot be opened will not be graded.**
- **Timing** Please start early. The problems are difficult and they can take hours to solve. The time you spend on finding the proof can be much longer than the time to write. If you submit within one week of the deadline you will get half credit. Any submission after that will not receive any credit.
- **Collaboration Policy** Please check this page for the collaboration policy. You are **not** allowed to discuss homework problems in groups of more than 3 students. **Failure to adhere to these guidelines will be promptly reported to the relevant authority without exception.**

Problem 1 (Depth First Search). (15 points) Given the following graph (Figure 1), you are going to perform a Depth First Search.

When you have the option of choosing multiple vertices (in the main loop or when choosing the next vertex inside the recursive call), always choose the vertices with smaller indices first. That is, think of the main loop as

```
FOR i = 1 to 7
    DFS_visit(i)
```

For enumerating the edges, think of

```
For v = 1 to 7
    IF (u,v) is an edge and v is not visited THEN
        DFS_visit(v)
```

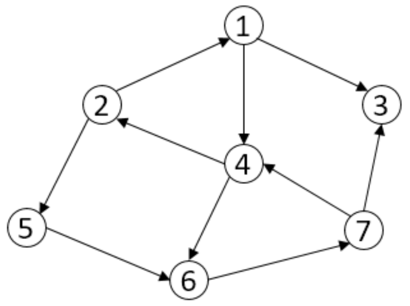

Figure 1: Graph for Problem 1

(a) (5 points) List the edges of DFS tree.

(b) (5 points) List the pre-order and post-order.

(c) (5 points) For all the non-tree edges, classify them into forward edges, back edges and cross edges.

Problem 2 (Graph Traversal)**.** (20 points) Recall that given a graph G, and a rooted tree T whose edges are a subset of the edges of the graph, a forward edge (blue edge in Figure 2) is an edge (u, v) of the graph where u is an ancestor of v (and u is not the direct parent of v). A cross edge (red edge in Figure 2) is an edge (u, v) of the graph such that u is not in the subtree rooted at v, and v is also not in the subtree rooted at u.

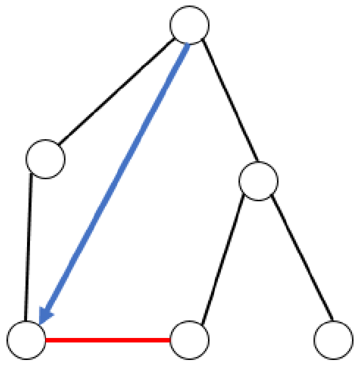

Figure 2: Black edges are tree edges, red edge is a cross edge and blue edge is a forward edge.

(a) (10 points) Prove that a DFS tree on an undirected graph does not have any cross edges.
(b) (10 points) Prove that a BFS tree does not have any forward edges.

Problem 3 (Store Locations)**.** (25 points) There are n neighborhoods in North Carolina. You are given a map where neighborhoods are vertices and adjacent neighborhoods are connected by an edge. Suppose we would like to open some stores in North Carolina, so that everyone can either go to a store in his/her neighborhood, or a store in an adjacent neighborhood. However due to budget problems we can only open at most $n/2$ stores.

There are at least two neighborhoods, and the graph on neighborhoods is connected.

(a) (15 points) Design an algorithm that finds at most $\lfloor n/2 \rfloor$ neighborhoods, such that if those neighborhoods have a store, everyone can find a store either in their neighborhood or in at least one of the adjacent neighborhoods.

(Hint: Use one of the graph traversal algorithms that we discussed.)

(b) (10 points) Prove the correctness of your algorithm.