Problem Set 8

Assigned: Apr. 2
Due: Apr. 9

Homeworks submitted electronically should be emailed to Changle Wang cl.wang@nyu.edu

Problem 1.

Let $G$ be a DAG. A vertex in $G$ is a sink if it has no outarcs. A forward path from vertex $U$ is a path that ends in a sink. Vertex $V$ is a terminus of vertex $U$ if $V$ is a sink and there is a path from $U$ to $V$.

A. Construct an algorithm $\text{NumForPaths}(G)$ that computes the number of forward paths from every node in DAG $G$ in linear time. If $U$ is itself a sink, then $\text{NumForPath}(G)[U]$ should be 1.

B. Construct an algorithm $\text{NumTerminus}(G,U)$ that computes the number of terminuses for vertex $U$ in DAG $G$ in linear time.

For instance in the following graph $G$

```
A -> B -> D
B -> E
C -> E
```

$\text{NumForPaths}(G)[A] = 3$: $A \rightarrow B \rightarrow D$; $A \rightarrow B \rightarrow E$; and $A \rightarrow C \rightarrow E$

$\text{NumTerminus}(G,A) = 2$: D and E.

Problem 2

(Siegel Ex. 7.16). To see how essential marking is for graph traversal, consider the application of depth-first-traversal on a DAG $G$ where node marking is not used. That is, the DFS code is rewritten in the form

```
procedure DFS(v) {
  for (each outarc v --> w) DFS(w)
}
```

Consider the complete DAG on $n$ vertices; that is, the vertices are numbers $1 \ldots n$ and there is an arc from $i$ to $j$ for every pair $i < j$. What is the running time of this modified DFS on that graph?
Problem 3

(Siegel Ex. 7.21). Write an algorithm that takes a DAG $G$ as input and prints out all the possible topological sorts of $G$. For instance, given the graph in problem 1, the algorithm would output

A, B, C, D, E
A, B, C, E, D
A, B, D, C, E
A, C, B, D, E
A, C, B, E, D

It should print out each sort only once. Your algorithm does not have to produce the sorts in this order.