CS/ECE 374 FALL 2018	Zhe Zhang (zzhan157@illinois.edu)
Homework 7 Problem 2	Ray Ying (xinruiy2@illinois.edu)
	Anqi Yao (anqiyao2@illinois.edu)

Let $G = (V, E)$ be a directed graph.

- Describe a linear-time algorithm that outputs all the nodes in G that are contained in some cycle. More formally you want to output

$$S = \{v \in V \mid \text{there is some cycle in } G \text{ that contains v}\}.$$

- Describe a linear time algorithm to check whether there is a node $v \in V$ such that v can reach every node in V. First solve the problem when G is a DAG and then generalize it via the meta-graph construction.

No proofs necessary but your algorithm should be clear. Use known algorithms as black boxes rather. In particular the linear-time algorithm to compute the meta-graph is useful here.

Solution:

1. For finding all vertices that contains in some cycle is similar to the question in class for finding all the SCCs for the graph. By the definition of SCCs, in a diagraph, every vertex in the cycle is strongly connected to all other vertex in the cycle, which means that a cycle will be a SCC or a subset of SCC, but a SCC is always a cycle. Also, the SCCs except these SCCs that only contain one vertex are all cycles. The algorithm in class that use the DFS to find all the SCCs will take $O(|V| + |E|)$. The difference between this question only lies in that the each time we find a SCC in the DFS, we will have to evaluate if the number of items in SCC is greater than 1. If it is, we will push the node in this SCC to the set S. Since DFS will only go to each node once by definition, the running time for still be linear $O(|V| + |E|)$.

2. The question asks about if there is a vertex $v \in V$ that can reach every node in V. In a DAG,(When there is at least 2 vertices), there is a topological ordering, which gives one vertex the highest order. In class, we discussed that using DFS, the running time for finding the topological sort is $O(|V| + |E|)$. After finding the topological sort, we will again apply DFS on the highest ranking vertex in the sort, to check if all the vertices is reached(If reach a new vertex, we add one to the count, if in the end the count is less than the total number of vertex, return false and stating no vertex can reach all vertices).
   If the graph is any digraph, what we do is first we using DFS to find the SCCs in the graph and draw the meta-graph. The running time for this process will be $O(2 * (|E| + |V|))$, since we at mostly will have $(|E| + |V|)$ number of components. The meta-graph is definitely DAG since otherwise, the SCCs are wrong(Piazza said the SCCs are the largest SCC, the subset of a SCC is not a SCC). For every SCC, we can just choose one arbitrary vertex in that SCC, in this case, we will have one vertex represent for that SCC. Using DFS for finding the topological sort in the orginal graph with the chosen vertices in the meta-graph and check if all vertices is reached. Since we are not asking to find all vertices that can reach

other vertex but just stating if such vertex exists, we can simply return true or false at this stage. This will take $O(|E|+|V|)$. So every step is linear time and the sum of all steps are $O(3*(|E|+|V|))$, implies the running time to be $O(|V|+|E|)$.

■