CSCE-629 Analysis of Algorithms

Spring 2019

Instructor: Dr. Jianer Chen	**Teaching Assistant:** Qin Huang
Office: HRBB 315C	**Office:** HRBB 315A
Phone: 845-4259	**Phone:** 402-6216
Email: `chen@cse.tamu.edu`	**Email:** `huangqin@email.tamu.edu`
Office Hours: T,Th 11:00 am–12:30 pm	**Office Hours:** MWF 3:00 pm–4:00 pm

Assignment # 4

(Due April 4, 2019)

1. Another way to perform topological sorting on a directed acyclic graph $G = (V, E)$ is to repeatedly find a vertex of in-degree 0, output it, and remove it and all of its outgoing edges from the graph. Develop an $O(|V| + |E|)$-time algorithm using this approach. Your algorithm should also be able to tell when the input graph has cycles.

2. Let G be a directed graph with strongly connected components C_1, C_2, $\ldots$, C_k. The *component graph* G^c for G is a directed graph of k vertices w_1, w_2, $\ldots$, v_k such that there is an edge from w_i to w_j in G^c if and only if there is an edge from some vertex in C_i to some vertex in C_j. Develop an $O(|V|+|E|)$-time algorithm that on a given directed graph $G = (V, E)$ produces the component graph G^c for G. Make sure that there is at most one edge between two vertices in the component graph G^c.

3. Given a linear-time algorithm that takes as input a directed acyclic graph G and two vertices s and t, and returns the number of simple paths from s to t in G. Your algorithm needs only to count the simple paths, not list them. Note that different paths from s to t may share common vertices.