1 Graph algorithms

1.1 Multigraph to simple graph

Given an adjacency-list representation of a multigraph $G = (V, E)$, describe an $O(V + E)$-time algorithm to compute the adjacency-list representation of the equivalent undirected graph ($H = (V, E')$), where E' consists of the edges in E with all multiple edges between two vertices replaced by a single edge and with all self-loops removed.

1.2 Neighborhood degrees

For each node u in an undirected graph $G = (V, E)$, let $d_2[u]$ be the sum of the degrees of u's neighbors. Show how to compute the entire array $d_2[\,]$ in linear time, given G in adjacency-list format.

1.3 Cyclic edge

Design a linear-time algorithm which, given an undirected graph G and a particular edge e in it, determines whether G has a cycle containing e.

1.4 Curriculum

Suppose a CS curriculum consists of n courses, all of them mandatory. The prerequisite graph G has a node for each course, and an edge from course v to course w if and only if v is a prerequisite for w. Propose an algorithm that works directly with this graph representation, and computes the minimum number of semesters necessary to complete the curriculum. Assume that a student can take any number of courses in one semester. The running time of your algorithm should be linear.

1.5 Reachability

Give an efficient algorithm which takes as input a directed graph $G = (V, E)$, and determines whether or not there is a vertex $s \in V$ from which all other vertices are reachable.

1.6 s-t path in DAG

Give an efficient algorithm that takes as input a directed acyclic graph $G = (V, E)$ and two vertices $s, t \in V$, and outputs the number of different directed paths from s to t in G.