Homework 2

Deadline: 14:20 on May 28, 2025

Exercise 1. (10 pts) Consider an undirected graph $G = (V,\, E)$. Assume that for every vertex $v \in V$, the edges of v are already stored in an adjacency list. Please design an $O(|V|)$-time algorithm that determines whether G contains a cycle. Note that to attain the $O(|V|)$ time, your algorithm will not necessarily check all the edges in E.
Hint: You may modify the DFS algorithm. Beware that if G contains no cycle, $|E| < |V|$.

Exercise 2. (10 pts) Consider an undirected, weighted, connected graph $G = (V, E, w)$. All edge weights in E are integers in the range from 1 to W with $|V| \le |W| \le |V|^{100}$, and the value of W is given. Please analyze how fast Kruskal's algorithm can be.
Hint: Radix-Sort takes $O\big(\frac{b}{r} \cdot (n + 2^r)\big)$ time where b the number of bits for an integer and r is a user-defined parameter. The operations for disjoint sets take $O\big(|E| \cdot \alpha(|V|)\big)$ time.

Exercise 3. (15 pts) Borůvka's algorithm in 1926 is the first algorithm for computing the minimum spanning tree and named after a Czech mathematician Otakar Borůvka. The main idea is to include, for every vertex, its lightest edge. Consider an undirected, weighted, connected graph $G = (V,\, E,\, w)$ and assume that all edges in E have different weights. Let E' be the set of the lightest edges of all vertices, i.e., $E' = \{(a,\, b) \mid \forall a \in V, (a,\, b) = \arg\min_{(a,\, c) \in E} w(a,\, c)\}$.
Please prove that the edges in E' belong to every minimum spanning tree.

Exercise 4. (10 pts) Consider a positive integer δ and a weighted directed graph $G = (V,\, E)$ with a source $s \in V$ and a target $t \in V$. Further assume that the weight of every edge in E is a nonnegative integer and the distance from s to t is at most δ. Design an efficient algorithm for finding a shortest path from s to t in $O\left(\delta + |V| + |E|\right)$ time.
Hint: Based on the known value of δ, you can avoid using a heap or a Fibonacci-heap.

Exercise 5. (15 pts) Consider a directed, weighted graph $G = (V, E, w)$ without any negative cycle. As a convention for the all-pairs shortest paths problem, let $V = \{1,\, 2, \ldots,\, n\}$, let $D = \{d_{ij}\}$ be the distance matrix and the let $\Pi = \{\pi_{ij}\}$ be the predecessor matrix. More precisely, d_{ij} is the length of a shortest path from i to j, and π_{ij} is the predecessor of j along a shortest path from i to j. Please explain how to compute D from Π and how to compute Π from D.
You have to analyze the two time complexities.

Exercise 6. (15 pts) Consider a weighted directed graph $G = (V, E)$.

1. Assume that for every ordered pair of vertices $u, v \in V$, there exists a shortest path from u to v using $O(\log n)$ edges. Please adapt the matrix-multiplication based algorithm (named FASTER-APSP in the text book) to this assumption and analyze its running time.

2. Assume that G contains at least one negative cycle. Design an efficient algorithm that finds a negative cycle from the output of the Floyd-Warshall algorithm (i.e., the distance matrix D and the predecessor matrix Π), and analyze its running time.

Exercise 7. (15 pts) You are given a directed graph $G = (V, E, w)$ with $w : E \mapsto \mathbb{R}_+$, a source $s \in V$ and a target $t \in V$. Two paths from s to t are called *vertex-disjoint* if they don't share any intermediate vertex, i.e., excluding the two endpoints s and t. Please design an efficient algorithm for finding a maximum set of vertex disjoint paths from s to t.
Hint: You can design a flow network by modifying G, e.g., modify all the vertices in $V \setminus \{s, t\}$.

Exercise 8. (10 pts) Given an integer δ and a weighted graph $G = (V, E, w)$ with $w : E \mapsto \mathbb{N}$, the *weighted longest path* problem is to decide if G contains a simple path whose weight is at least δ. Recall that a simple path does not visit the same vertex twice and the weight of a path is the sum of weights of its edges. Assume that $\mathcal{A}$ is an algorithm for the weighted longest path problem with running time $O(|V|^2 \cdot |E|^9 \cdot \delta)$. Please answer if $\mathcal{A}$ is a polynomial-time algorithm and justify your answer.

Exercise 9. (Not graded) Given an undirected graph $G = (V, E)$, an *independent set* is a subset $C \subseteq V$ such that for every two vertices $u, v \in C$, $(u, v) \notin E$.

a Please define a decision problem for independent sets.

b Please prove that the decision problem belongs to NP.

c Please prove that the decision problem is NP-Hard.

Hint: You can assume that the *vertex cover* problem is NP-Complete. Recall that for an undirected graph $G = (V, E)$, a vertex cover is a subset $C \subseteq V$ such that for every edge $(u, v) \in E$, $u \in C$ or $v \in C$.

Exercise 10. (Not graded) Prove that the vertex cover problem can be reduced from the 3-CNF-SAT problem in polynomial time. You are not allowed to use the transitivity of the reduction.