

IMPORTANT: Write each problem on a separate sheet of paper. Write your name on each sheet. (Why? To speed up the grading we will use parallelization—each problem will be graded by a different TA. At the beginning of the class (on the date when the homework is due) there will be a separate pile for each problem.)

Homework 2 due: Oct. 1 (Thursday); collected before class in B&L 109.

Problem sessions before Homework 2 is due:

Sep. 28 (Monday) 6:30pm - 7:30pm in Goergen 108

Sep. 29 (Tuesday) 6:00pm - 7:00pm in Goergen 108

Sep. 30 (Wednesday) 6:15pm - 7:15pm in Goergen 108

Sep. 30 (Wednesday) 7:30pm - 8:30pm in Goergen 108

1 Homework - solve and turn in

2.1 (due Thursday, Oct 1) Let $G = (V, E)$ be a directed graph (digraph) given in the following adjacency-list representation: for each vertex $v \in V$ we have a linked list of out-neighbors of v . Assume that $V = \{1, \dots, n\}$; let $m := |E|$.

1. Write pseudocode for a procedure which outputs an adjacency-list representation of the reverse digraph (i. e., G with each edge reversed). The procedure should run in time $O(m + n)$.
2. Write pseudocode for a procedure which outputs the adjacency-list representation of G in which the out-neighbors of each vertex are listed in the increasing order. The procedure should run in time $O(m + n)$.
3. Write pseudocode for a procedure which checks if G is undirected (i. e., the reverse of every $e \in E$ is also in E). The procedure should run in time $O(m + n)$.

2.2 (due Thursday, Oct 1) Let $G = (V, E)$ be an directed graph; let $n := |V|$, $m := |E|$. Assume that G is not strongly connected. Give an $O(m + n)$ algorithm that decides whether there exist $u, v \in V$ such that adding edge (u, v) to G makes it strongly connected.

2.3 (due Thursday, Oct 1) We have a road network consisting of one-way streets. There are a few proposals for a new one-way road to be built. You want to vote for the proposal that would result in the shortest distance between your home and your office. How can you, quickly, figure out the proposal to vote for? We now formalize the problem.

Let $G = (V, E, w)$ be a directed graph with positive edge weights; let $n := |V|$, $m := |E|$. Let $s, t \in V$ be two distinct vertices (s is your home, t is your office). There are k proposals $(a_1, b_1, c_1), \dots, (a_k, b_k, c_k)$, where $a_i, b_i \in V$ and $c_i \in \mathbb{R}_+$ ((a_i, b_i, c_i) proposes connecting location a_i to location b_i with a road of length c_i).

Let G_i be the graph obtained from G by adding edge (a_i, b_i) with weight c_i (note that G_i has $m + 1$ edges (for each $i \in \{1, \dots, k\}$)). We want to find which of G_1, \dots, G_k has the shortest distance from s to t . Describe an $O(m + n \log n + k)$ algorithm for the problem (no pseudocode is needed, just a clear description of the algorithm; you can use any algorithm that was mentioned in the class or in the book).

2 Bonus Homework - solve and turn in

2.4 (due Thursday, Oct 1) Let $G = (V, E, w)$ be an undirected graph with positive edge weights; let $m := |E|$, $n := |V|$. Let $(a_1, b_1, c_1), \dots, (a_k, b_k, c_k)$ be a collection of triples, where $a_i, b_i \in V$ and $c_i \in \mathbb{R}_+$ (for $i \in \{1, \dots, k\}$). Let G_i be the graph obtained from G by adding edge $\{a_i, b_i\}$ with weight c_i . We want to find the cost of the minimum weight spanning tree of G_i , for each $i \in \{1, \dots, k\}$. (Thus the output of our procedure is k numbers.)

Describe an $O(n^2 + k)$ algorithm for the problem. Clearly describe your algorithm in words, give a pseudocode, and argue why the running time is $O(n^2 + k)$ (if you have a faster algorithm that is, of course, OK). You can use any algorithm that was mentioned in the class or in the book (without writing the pseudocode for that algorithm).

3 Additional problems from the book

Try to solve the following problems. We will go over the ones that you choose in the problem sessions.

- 3.1, 3.2, 3.3, 3.4, 3.6, 3.7, 3.8, 3.9, 3.11, 3.12, 3.13, 3.15, 3.18, 3.19, 3.21, 3.22, 3.23, 3.24, 3.25, 3.26, 3.27,
- 4.1, 4.2, 4.3, 4.4, 4.5, 4.7, 4.8, 4.10, 4.11, 4.13, 4.15, 4.17, 4.18, 4.19, 4.20, 4.21,
- 5.1, 5.2, 5.3, 5.7.