Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Departement of Computer Science
Markus Püschel, David Steurer
Johannes Lengler, Gleb Novikov, Chris Wendler, Ulysse Schaller

23. November 2020

Algorithms & Data Structures — Exercise sheet 10 — HS 20

Exercise Class (Room & TA): _____

Submitted by: _____

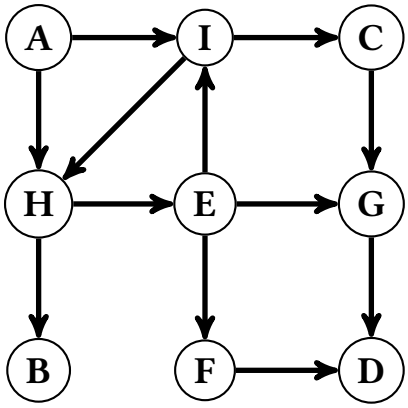
Peer Feedback by: _____

Points: _____

Submission: On Monday, 30 November 2020, hand in your solution to your TA *before* the exercise class starts. Exercises that are marked by * are challenge exercises. They do not count towards bonus points.

Exercise 10.1 *Depth-First Search.*

Execute a depth-first search (Tiefensuche) on the following graph starting from vertex A. Use the algorithm presented in the lecture. When processing the neighbors of a vertex, process them in alphabetical order.



a) Mark the edges that belong to the depth-first tree (Tiefensuchbaum) with a "T" (for tree edge).

b) For each vertex, give its *pre*- and *post*-number.

c) Give the vertex ordering that results from sorting the vertices by pre-number. Give the vertex ordering that results from sorting the vertices by post-number.

d) Mark every forward edge (Vorwärtskante) with an "F", every backward edge (Rückwärtskante) with an "B", and every cross edge (Querkante) with a "C".

e) Does the above graph have a topological ordering? How can we use the above execution of depth-first search to find a directed cycle?

f) Draw a scale from 1 to 18, and mark for every vertex v the interval I_v from pre-number to post-number of v. What does it mean if $I_u \subset I_v$ for two different vertices u and v?

g) Consider the graph above with the edge from I to H removed. How does the execution of depth-first search change? Which topological sorting does the depth-first search give? If you sort the vertices by *pre-number*, does this give a topological sorting?

Exercise 10.2 *Completing a directed acyclic graph* **(1 point)**.

Let $G = (V, E)$ be a directed acyclic graph on n vertices. Show that it is possible to add (directed) edges to G to obtain a directed acyclic graph $G' = (V, E')$ such that:

- $E \subseteq E'$, i.e. G is a subgraph of G',
- for every pair of distinct vertices $u, v \in V$, either $(u, v) \in E'$ or $(v, u) \in E'$, i.e. every pair of vertices is connected with an edge (that can go in either of the two directions).

Exercise 10.3 *Mountain bike tour* **(2 points)**.

You plan a mountain bike tour for you and your best friend. Your friend only bikes downhill and asks you to find the longest possible tour starting from the top of Pilatus.

Assume that you are given a map with the information about the tours that start from the top of Pilatus. Concretely, the map contains all junctions (including the top of Pilatus) and their absolute heights. Moreover, if there is a trail from junction j_1 to junction j_2 that does not traverse any other junctions, the map contains the length of this trail (you can assume that if such a trail exists, then it is unique). You can also assume that if there exists such a trail from junction j_1 of height h_1 to junction j_2 of height h_2 and $h_2 < h_1$, then the whole trail from j_1 to j_2 only goes downhill.

Note that since your friend only bikes downhill, she cannot visit junction j_2 of height h_2 safter junction j_1 of height h_1 if $h_2 \geq h_1$.

a) Model the problem as a graph problem on a directed graph. Describe the set of vertices, the set of edges and the edge weights in words. What is the graph problem corresponding to finding the longest downhill-only tour starting from the top of Pilatus?

b) Provide as efficient as possible *dynamic programming* algorithm that, given your graph G from a) determines the length of the longest downhill-only tour starting from the top of Pilatus.

Hint: *In this exercise you can assume that the directed graph is represented by a data structure that allows you to traverse the direct successors and direct predecessors of a vertex u in time $\mathcal{O}(deg_+(u))$ and $\mathcal{O}(deg_-(u))$ respectively, where $\deg_-(u)$ is the in-degree of vertex u and $deg_+(u)$ is the out-degree of vertex u.*

Hint: *The problem gets a lot easier to solve when you start by topologically sorting the vertices of your graph.*

Dimensions of the table:

Meaning of a table entry (in words):

Computation of an entry (initialization and recursion):

Order of computation:

Computing the output:

Running time in $\mathcal{O}$-notation in terms of $|V|$ and $|E|$: