1 Homework 2 Due April 24, 9:20 in canvas

Acyclic Graphs and Topological Order

Homework 2.01

Consider the graph shown in Figure 1 below, except that the edges (G, A) and (C, D) are deleted.

Carry out the topological-order algorithm described in lecture and shown in scanned handouts from the lectures. You may build a stack instead of keeping track of numbers. What reverse topological order is found if vertices are traversed in alphabetical order and adjacency lists are in alphabetical order (A, B, C, D, E, F, G, H)?

Homework 2.02

Like Homework 2.01 except that vertices are traversed in **reverse** alphabetical order and adjacency lists are in **reverse** alphabetical order. Is the topological order the same in both cases?

Asymptotic Analysis

Homework 2.03

Let $G = (V, E)$ be a directed graph with n vertices and m edges. It is known that in `dfsTrace` of G the function `dfs` is called n times, once for each vertex.

It is also seen that `dfs` contains a loop whose body gets executed while visiting v once for each vertex w adjacent to v; that is the body gets executed once for each edge (v, w). In the worst case there are n adjacent vertices.

What do these facts imply about the best asymptotic order (big O and/or big Θ) we can derive in terms of n and m about the worst-case time for `dfsTrace`? Explain your answer briefly.

Strongly Connected Components

Homework 2.04

Give an example of a directed graph for which the SCC algorithm gives the wrong SCCs if the second depth-first search is performed on G instead of on G^T. Assume the finishing-time stack is still used by the second dfsSweep. A graph of 5–6 vertices shoud be sufficient to make this counter-example.

Homework 2.05

Give an example of a directed graph for which the SCC algorithm gives the wrong SCCs if the second depth-first search is performed on G^T, but second dfsSweep traverses the finishing-time stack from bottom to top instead of top to bottom.

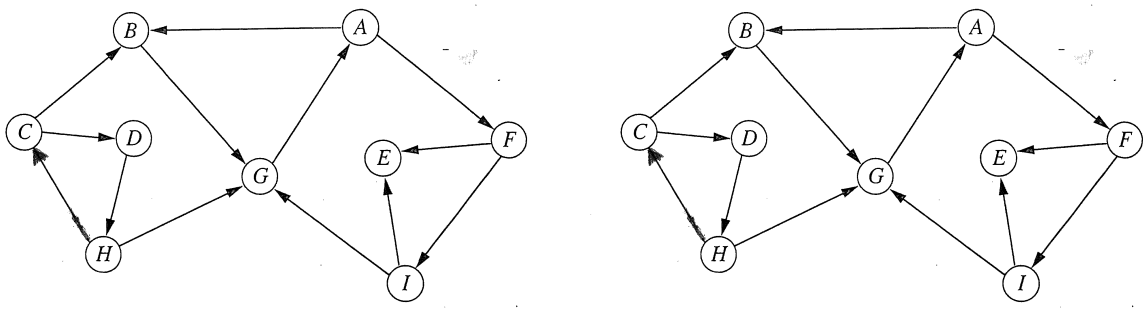

Figure 1: Digraph (two copies) for homeworks 2.06–2.09. Think about why two copies are shown.

Homework 2.06–2.09

Find the strongly connected components of the graph shown twice in Figure 1 by carefully following the steps of the SCC algorithm due to Sharir, as presented in lectures and found in the optional texts (excerpts in `Handouts`). You will make two assumptions about the order of traversals, as stated below.

Be sure to show any important data structures computed by the SCC algorithm.

Indicate the SCCs and where they were found, preferably by circling them in the appropriate picture.

Also show the discovery times and finishing times. Also heavy-line the tree edges and circle the tree roots of both depth-first searches.

2.06: Traverse the vertices in alphabetical order for the first DFS and assume the adjacency lists are also in alphabetical order (A, B, C, D, E, F, G, H).

2.07: Traverse the vertices in **reverse** alphabetical order for the first DFS and assume the adjacency lists are also in reverse alphabetical order.

2.08: Identify the **SCC leaders** found in 2.06 and 2.07. Are they the same? Briefly explain how you identified them.

2.09: Finally, draw the component graph(s), also called condensation graphs, for 2.06 and 2.07. Are they the same?

Homework 2.10

Recall that in the SCC algorithm the **leader** of an SCC is the first vertex in that SCC to be **discovered** by a depth-first search.

Explain how the White Path Theorem and the Nesting of Descendant Intervals Lemma imply that each SCC has the same leader in the first DFS (on G) and the second DFS (on G^T).