Homework Assignment #9

Due Time/Date

2:10PM Tuesday, December 22, 2020. Late submission will be penalized by 20% for each working day overdue.

Note

Please write or type your answers on A4 (or similar size) paper. Drop your homework by the due time in Yih-Kuen Tsay's mail box on the first floor of Management College Building 2. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

There are five problems in this assignment, each accounting for 20 points. (Note: problems marked with "(X.XX)" are taken from [Manber 1989] with probable adaptation.)

1. (7.16 modified)

   (a) Run the strongly connected components algorithm on the directed graph shown in Figure 1. When traversing the graph, the algorithm should follow the given DFS numbers. Show the *High* values as computed by the algorithm in each step.

   (b) Add the edge $(6, 8)$ to the graph and discuss the changes this makes to the execution of the algorithm.

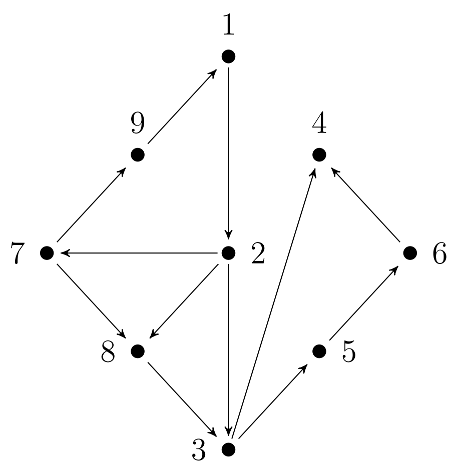

Figure 1: A directed graph with DFS numbers

2. (7.88) Let $G = (V, E)$ be a directed graph, and let T be a DFS tree of G. Prove that the intersection of the edges of T with the edges of any strongly connected component of G form a subtree of T.

3. Consider the algorithm discussed in class for determining the strongly connected components of a directed graph. Is the algorithm still correct if we replace the line "$v.high := \max(v.high, w.DFS\_Number)$" by "$v.high := \max(v.high, w.high)$"? Why? Please explain.

4. Consider designing an algorithm by dynamic programming to determine the length of a longest common subsequence of two strings (sequences of letters). For example, "abbcc" is a longest common subsequence of "abcabcabc" and "aaabbbccc", and so is "abccc".

   (a) Formulate the solution using recurrence relations.

   (b) Present the algorithm in suitable pseudocode, based on the previous recursive formulation. What is the time complexity of your algorithm?

5. Consider designing by dynamic programming an algorithm that, given as input a sequence of distinct numbers, determines the length of a longest increasing subsequence in the input sequence. For instance, if the input sequence is $1, 3, 11, 5, 12, 14, 7, 9, 15$, then a longest subsequence is $1, 3, 5, 7, 9, 15$ whose length is 6 (another longest subsequence is $1, 3, 11, 12, 14, 15$).

   (a) Formulate the solution using recurrence relations.

   (b) Present the algorithm in suitable pseudocode, based on the previous recursive formulation. What is the time complexity of your algorithm?