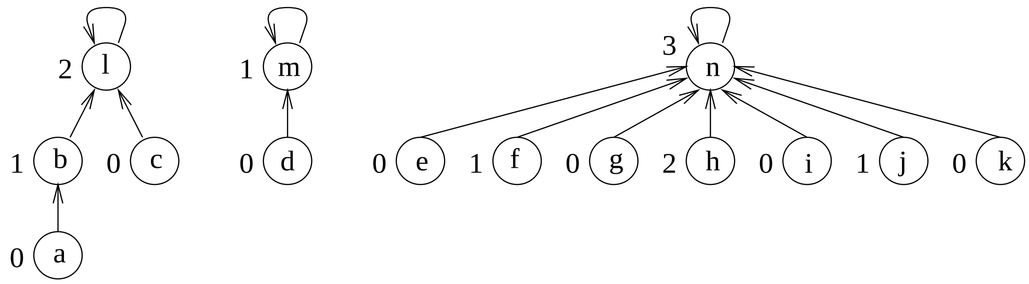
Analysis of Algorithms: Assignment 6

Due date: March 27 (Monday)

If you submit the assignment by 12:30pm on March 22 (Wednesday), then you will earn 2 bonus points toward your grade for this assignment (not toward the final grade for the course). If you submit it by 12:30pm on March 24 (Friday), you will get 1 bonus point.

Problem 1 (3 points)
Suppose we apply the CONNECTED-COMPONENTS algorithm to an undirected graph G, with vertices $G[V] = \{a, b, c, d, e, f, g, h, i, j, k\}$, and its edges $E[G]$ are processed in the following order: $(e, g), (a, d), (i, k), (c, g), (b, f), (b, h), (f, k), (a, k), (f, h), (d, i)$. Using Figure 22.1 in the textbook as a model, illustrate the steps of CONNECTED-COMPONENTS on this graph.

Problem 2 (3 points)
Consider the disjoint-set forest shown below, where numbers are the ranks of elements, and suppose that you apply three successive operations to this forest: FIND-SET(a), UNION(l, d), and UNION(d, e). Give a picture of the disjoint forest after each of these operations; thus, you need to draw three different pictures.



Problem 3 (4 points)
The *transpose* of a given graph G is the result of reversing all edges in G. Thus, the transpose has the same vertices as G, and the same number of edges. For each edge (u, v) in G, the transpose includes the opposite edge (v, u).

Write an algorithm that computes the transpose of a given graph; both the initial graph and its transpose should be represented by adjacency lists. Give the time complexity (Θ-notation) of your algorithm.

Problem 4 (bonus)
This problem is optional, and it allows you to get 2 bonus points toward your final grade for the course. You cannot submit this bonus problem after the deadline.

Suppose that the rank of each node in a disjoint-set forest must be the exact height of the node, rather than an upper bound on the height. Then, FIND-SET has to change the ranks of the nodes on the compressed path.

Describe a modified representation of the disjoint-set forest, which supports this operation, and the corresponding modifications of MAKE-SET, UNION, and FIND-SET. What is the time complexity of the resulting implementation, in terms of m and n?