Homework 05

CS 624, 2024 Spring

1. Problem 22.1-1 (p592).

   > Given an adjacency-list representation of a directed graph, how long does it take to compute the out-degree of every vertex? How long does it take to compute the in-degrees?

2. Problem 22.2-1

   > Show the d and π values that result from running breadth-first search on the directed graph of Figure 22.2(a), using vertex 3 as the source.

3. Problem 22.2-2

   > Show the d and π values that result from running breadth-first search on the undirected graph of Figure 22.3, using vertex u as the source.

4. Problem 22.2-3 (p602) — Note, corrected in 3rd printing.

   > Show that using a single bit to store each vertex color suffices by arguing that the BFS procedure would produce the same result if **line 18** were removed.

5. Problem 22.2-4 (p602)

   > What is the running time of BFS if we represent its input graph by an adjacency matrix and modify the algorithm to handle this form of input?

6. Problem 22.2-7 (p602)

   > There are two types of professional wrestlers: "babyfaces" ("good guys") and "heels" ("bad guys"). Between any pair of professional wrestlers, there may or may not be a rivalry. Suppose we have n professional wrestlers and we have a list of r pairs of wrestlers for which there are rivalries. Give an $O(n + r)$-time algorithm that determines whether it is possible to designate some of the wrestlers as babyfaces and the remainder as heels such that each rivalry is between a babyface and a heel. If it is possible to perform such a designation, your algorithm should produce it.

7. Problem 22.3-8 (p611)

   > Give a counterexample to the conjecture that if a directed graph G contains a path from u to v and if $u.d < v.d$ in in a depth-first search of G, then v is a descendant of u in the depth-first forest produced.

8. Problem 22.3-9 (p612)

   > Give a counterexample to the conjecture that if a directed graph G contains a path from u to v then any depth-first search must result in $v.d \leq u.f$.

9. Problem 22.5-1 (p620)

   > How can the number of strongly connected components of a graph change if a new edge is added?

   For each possibility, give an example that illustrates it.

10. Problem 22.5-3 (p620)

Professor Bacon claims that the algorithm for strongly connected components would be simpler if it used the original (instead of the transpose) graph in the second depth-first search and scanned the vertices in order of increasing finishing times. Does this simpler algorithm always produce correct results?