Algorithms Homework 8*

Fundamental techniques: Graph basics

Reading: GT Chapter 6

1. (4.2.27 Sedgewick Wayne) Explain why the following algorithm does not necessarily produce a topological order: Run BFS, and label the vertices by increasing distance to their respective sources. (Note: There may be more than one source vertex because they keep running BFS until all vertices are marked, see slides). (Note: To prove that a certain algorithm does not work, it's sufficient to show a counter-example).

2. (CLRS 22.1-3) The *transpose* of a digraph $G = (V, E)$ is the graph $G^T = (V, E^T)$, where $E^T = \{(v, u) \in VxV | (u, v) \in E\}$. In other words, G^T is G with all edges reversed. Describe efficient algorithms for computing G^T from G, for both adjacency-list and adjacency-matrix representation of G. Analysize the running times of your algorithms.

3. (CLRS 22.1-5) The *square* of a digraph $G = (V, E)$ is the graph $G^2 = (V, E^2)$ such that $(u, w) \in E^2$ if and only if for some vertex $v \in V$, both $(u, v) \in E$ and $(v, w) \in E$. That is, G^2 contains an edge from u to w whenever G contains a path with exactly two edges from u to w. Describe efficient algorithms for computing G^2 from G, for both adjacency-list and adjacency-matrix representation of G. Analysize the running times of your algorithms.

4. (CLRS 22.4-3) Give an algorithm that determines whether or not a given undirected graph contains a cycle. Analyze carefully the running time of your algorithm — ideally you would have an algorithm that runs in $O(V)$ time (not the usual $O(V + E)$).

5. (CLRS 22-4) *Reachability.* Let $G = (V, E)$ be a digraph in which every vertex $u \in V$ is labeled with a unique label $L(u)$ from the set $\{1, 2, 3, ..., |V|\}$. For each vertex $u \in V$, let $R(u) = \{v \in V | u \text{ reaches } v\}$ be the set of vertices that are reachable from u. Define $\min(u)$ to be the vertex in $R(u)$ whose label is minimum, i.e. $\min(u)$ is the vertex v such that $L(v) = \min\{L(w) | w \in R(u)\}$. Give an $O(V + E)$ algorithm that computes $\min(u)$ for all vertices $u \in V$.

   Hint: Use BFS or DFS

*Collaboration is allowed, even encouraged, provided that the names of the collaborators are listed along with the solutions. Write up the solutions on your own.

6. **Extra credit** (CLRS 22.2-7) The *diameter* of a tree $T = (V, E)$ is given by:

$$\max_{u,v \in V} \delta(u, v)$$

where $\delta(u, v)$ is the shortest-path distance from u to v (by default the distance is the number of edges on a path). In other words, the diameter is the largest of all shortest-path distances in the tree. Give an efficient algorithm to compute the diameter of a tree, and analyze its running time.

Hint: Let x be a node in the tree, and $T(x)$ the sub-tree rooted at x. Express recursively the diameter of $T(x)$ function of the diameter of the children of x. Look for a relation to the height.