Algorithms Lab 8

The in-lab problems are to be solved during the lab time. Work with your team. You do not need to turn in your answers. Ask me for feedback.

The homework problem set is due in one week. Work with your team, but write your solutions individually. List the people with whom you discussed the problems clearly on the first page.

Grading guidelines[1]: A solution is evaluated for:

- correctness (does your algorithm solve the problem)
- analysis (specify space and time complexity)
- style (should look professional, neat, that means not messy, imagine this is your first assignment at Google and you explain it to your supervisor; leave space for us to write feedback)
- clarity (should be easy to understand)
- conciseness (avoid unnecessary explanations)

On a 1-to-10 scale:

- [10 points] The solution is clear, correct, neat and concise.
- [8-9 points] The solution contains a few mistakes, but they are mostly arithmetic or of little significance to the overall argument.
- [6-7 points] The solution hits on the main points, but has at least one logical gap.
- [4-5 points] The solution contains several logical mistakes, but parts of it are salvageable.
- [2-3 points] The solution is just plain wrong.
- [<2 points] No attempt is made at solving the problem.

[1] Based on guidelines at Williams, Swarthmore, Carleton, MIT

In lab

Work on the handout with applications of BFS/DFS from class.

Homework

1. (CLRS 22.2-3) Analyse BFS running time if the graph is represented by an adjacency-matrix.

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) *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 $u \in V$.

5. **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 among any two nodes in the tree. Give an efficient algorithm to compute the diameter of a tree, and analyze its running time.

Hint: Does the path that corresponds to the diameter (ie the longest path in the tree) have to go through the root of the tree? Why, or why not? Let x be a node in the tree, and $T(x)$ the sub-tree rooted at x. Express the diameter of $T(x)$ function of the children of x. Look for a relation to the height. First assume that the tree is binary, and perhaps that will make things a bit easier. Then you can extend it to arbitrary trees.