COMPSCI 311: Introduction to Algorithms **Fall 2019**

Homework 3

Released 10/10/2019 Due 10/22/2019 11:59pm in Gradescope

Instructions. You may work in groups, but you must write solutions yourself. List collaborators on your submission. Also list any sources of help (including online sources) other than the textbook and course staff.

If you are asked to design an algorithm, please provide: (a) the pseudocode or precise description in words of the algorithm, (b) an explanation of the intuition for the algorithm, (c) a proof of correctness, (d) the running time of your algorithm and (e) justification for your running time analysis.

Submissions. Please submit a PDF file. You may submit a scanned handwritten document, but a typed submission is preferred. Please assign pages to questions in Gradescope.

1. **(20 points) Restricted Roads** A set V of villages in a paranoid dictatorship is connected by a set of roads E to form a connected undirected graph G. Every road in E is labeled with a non-negative integer indicating the minimum level of security clearance needed to traverse that road legally. (So if a road is labeled "7", anyone cleared to level 7 or higher may use that road.)

   Give an algorithm that takes the labeled graph G as input and returns
   (i) the minimum level of clearance needed to be able to reach any of the villages in V from any other, and
   (ii) a minimal subset S of the roads that can be used with that clearance and which allow access to all villages. ("Minimal" means that if you remove any one road from S, there is at least one pair of villages that are no longer connected to one another.)

   Prove that your algorithm is correct and determine its big-O running time.

2. **(20 points) Recurrences** Give asymptotic upper and lower bounds in the following recurrences. Assume that $T(n)$ is bounded by a constant for $n \leq 2$. Make your bounds as tight as possible and justify your answers.

   a) $T(n) = T(n/2) + T(n/4) + 2n$

   b) $T(n) = \frac{1}{2}(T(n-1) + T(n-2)) + cn$, with $c > 0$

   c) $T(n) = \sqrt{n}T(\sqrt{n}) + n$

3. **(20 points) Three-way Sort** Consider the following Java-like code for a recursive sorting algorithm:

```
// sorts the items from L[i] through L[j]
void threewaysort(int[ ] L, int i, int j) {
  if (L[i] > L[j]) swap (i, j);
  if ((j - i + 1) > 2) {
    t = (j - i + 1)/3;
    threewaysort(L, i, j-t);
    threewaysort(L, i+t, j);
    threewaysort(L, i, j-t);
  }
}
```

   (a) Prove that this algorithm is correct, that is, that the call `threewaysort(L, 0, L.length-1)` actually has the side effect of making `L` sorted.

   (b) Let $f(n)$ be the running time of `threewaysort(L,i,j)` when $j - i + 1 = n$.
   Write a recurrence for $f(n)$.

   (c) Solve your recurrence to determine the worst-case running time of threewaysort on lists of size n. How does it compare to other sorting algorithms you know?

4. **(20 points) Weighted Path**. You are given a rooted binary tree (each node has at most two children). For a simple path p between two nodes in the tree, let m_p be the node on the path that is highest (closest to the root). Define the weight of a path $w(p) = \Sigma_{u \in p} d(u, m_p)$, where d denotes the distance (number of edges on the path between two nodes). That is, every node on the path is weighted by the distance to the highest node on the path.

   Design an algorithm that finds the maximum weight among all simple paths in the tree.

5. **(20 points) Maximum Tree Value** Consider an array of n positive integers $a_1 \dots a_n$. We'd like to construct a binary tree (where each non-leaf node has exactly two children) that has these numbers as leaves, left to right (equivalently, in inorder traversal). We compute a tree's value as follows: the value of a leaf is its number. Otherwise, the value of a tree is the sum of the values of its two subtrees, plus the product between the rightmost leaf of the left subtree and the leftmost leaf of the right subtree.

   Design an algorithm that computes the maximum value among all possible trees that we can build having the given numbers as leaves from left to right. The running time should be polynomial in n.

6. **(0 points).** How long did it take you to complete this assignment?