COMPSCI 311: Introduction to Algorithms **Fall 2018**

Homework 1

Released 9/6/2018 Due 9/20/2018 11:59pm in Gradescope

Instructions. You may work in groups, but you must individually write your solutions yourself. List your collaborators on your submission.

If you are asked to design an algorithm as part of a homework problem, please provide: (a) the pseudocode for 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. It will be very helpful if in your submission each question starts on a new page.

1. **(20 points) Stable Matching Running Time.** In class, we saw that the Propose-and-reject algorithm terminates in at most n^2 iterations, when there are n students and n colleges.

   (a) Construct an instance so that the algorithm requires only $O(n)$ iterations, and prove this fact.

   (b) Construct an instance so that the algorithm requires $\Omega(n^2)$ iterations (that is, it requires at least $c\,n^2$ iterations for some constant $c > 0$), and prove this fact.

   Your constructions for a) and b) should be possible for all values of n.

2. **(15 points) Streams and Junctions** (shorter rewording of K&T Ch 1 Ex 7)

   There are n input pipes and n output pipes, each of which can carry a data stream in one direction. At various points along it, each input pipe connects to each output pipe through a *junction* (a total of n^2 junctions). If a junction is *closed*, the data flows through it, staying on the same pipe. If a junction is *open*, the data goes from the input pipe to the output pipe, and continues to flow through the latter.

   Show that no matter how input and output pipes are connected through junctions, it is possible to open a set of junctions such that each data stream is switched from an input to an output pipe, and no two streams ever pass through the same junction. Give an algorithm to find such a valid switching.

3. **(15 points) Big-O and Big-Ω** For each function $f(n)$ below, find (1) the smallest *integer* constant H such that $f(n) = O(n^H)$, and (2) the largest *positive real* constant L such that $f(n) = \Omega(n^L)$. Otherwise, indicate that H or L do not exist. All logarithms are to base 2.

   (a) $f(n) = \frac{(n-1)n(n+1)}{2}$.

   (b) $f(n) = n(\log n)^2$

   (c) $f(n) = \sum_{k=0}^{\lceil \log n \rceil} \frac{n}{2^k}$.

   (d) $f(n) = \sum_{k=1}^{n} k(k+1)$.

   (e) $f(n) = 2^{(\log n)^3}$

4. **(15 points) Asymptotics** (variation from K&T)

   Given an array A of n integers, fill in an $n \times n$ array B such that for all $i < j$, $B[i,j]$ is the alternating sum $A[i] - A[i+1] + \ldots + (-1)^{j-i}A[j]$. For $i \geq j$ the value of $B[i,j]$ can be left as is.

```
for $i = 1, 2, \ldots, n$
    for $j = i+1, \ldots, n$
        compute $A[i] - A[i+1] + \ldots + (-1)^{j-i}A[j]$.
        store in $B[i,j]$.
```

(a) What is the running time of this algorithm as a function of n? Specify a function f such that the running time of the algorithm is $\Theta(f(n))$.

(b) Design and analyze a faster algorithm for this problem. You should give an algorithm with running time $O(g(n))$, where $\lim_{n\to\infty} g(n)/f(n) = 0$.

5. **(15 points) DFS and BFS. K&T Ch 3 Ex 6**

   Suppose we have a connected graph $G = (V, E)$ and a vertex $u \in V$. If we run DFS from u, we obtain a tree T. Suppose that if we run BFS from u we obtain exactly the same tree T. Prove that $G = T$.

6. **(20 points) Reachability** (Cormen, Leiserson, Rivest, Stein, problem 22-4)

   Let $G = (V, E)$ be a directed graph in which each vertex $u \in V$ is labeled with a unique integer $L(u)$ from the set $\{1, 2, \dots, |V|\}$. For each vertex $u \in V$, let $R(u) = \{v \in V : u \leadsto v\}$ be the set of vertices that are reachable from u. Define $\min(u)$ to be the vertex in $R(u)$ whose label is minimal, i.e., $\min(u)$ is the vertex v such that $L(v) = \min\{L(w) : w \in R(u)\}$. Give an $O(V + E)$-time algorithm that computes $\min(u)$ for all vertices $u \in V$.

7. **(0 points).** How long did it take you to complete this assignment?